Titre: Title:	Mécanismes d'attention pour les modèles convolutifs dans le cadre de la prédiction de trajectoires
Auteur: Author:	Laurent Boucaud
Date:	2019
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Boucaud, L. (2019). Mécanismes d'attention pour les modèles convolutifs dans le cadre de la prédiction de trajectoires [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/3951/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/3951/
Directeurs de recherche: Advisors:	Daniel Aloise, & Nicolas Saunier
Programme: Program:	Génie informatique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Mécanismes d'attention pour les modèles convolutifs dans le cadre de la prédiction de trajectoires

LAURENT BOUCAUD
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Août 2019

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Mécanismes d'attention pour les modèles convolutifs dans le cadre de la prédiction de trajectoires

présenté par **Laurent BOUCAUD**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Michel DESMARAIS, président
Daniel ALOISE, membre et directeur de recherche
Nicolas SAUNIER, membre et codirecteur de recherche
James GOULET, membre

DÉDICACE

À mes parents,
pour leur soutien...

REMERCIEMENTS

J'aimerais d'abord remercier mon directeur de recherche Daniel Aloise et mon co-directeur Nicolas Saunier pour l'intérêt qu'ils ont pu porter à mon sujet de recherche.

J'aimerais également remercier l'Institut de valorisation des données (IVADO) pour le financement de mon projet de recherche.

J'aimerais également remercier mes parents pour leur soutien sans faille et leurs bons conseils ainsi que Kristina qui m'a permis de conserver ma santé mentale.

Enfin, merci à la fine équipe Clara, Kim, Leandro, Rodrigo, Thiago et Théo qui ont su briser la monotonie du labo.

RÉSUMÉ

Cette maîtrise porte sur le problème de prédiction des mouvements des usagers de la route de différents types (piétons, cyclistes, automobilistes...) interagissant ensemble au sein d'intersections. Prédire les déplacements futurs d'agents présente un intérêt dans divers domaines comme les voitures autonomes et les analyses de sécurité routière. Si diverses approches ont été proposées pour traiter ce problème, l'explosion récente des performances des méthodes d'apprentissage profond dans des domaines comme la vision par ordinateur ou le traitement du langage naturel a conduit à son utilisation pour la tâche de prédiction de trajectoires. Un intérêt majeur de l'apprentissage profond étant de ne pas devoir définir manuellement les caractéristiques pertinentes à utiliser pour réaliser une prédiction, mais laisser les modèles l'apprendre automatiquement. Des modèles simples ont été proposés, prédisant la future trajectoire d'un agent en se basant uniquement sur son déplacement passé. Ces modèles peuvent être considérés comme naïfs puisque n'ayant pas conscience de l'environnement dans lequel l'agent évolue. Cet environnement est constitué tant d'une dimension sociale, c'est-à-dire l'influence mutuelle des agents sur leurs déplacements respectifs, que d'une dimension spatiale, c'est-à-dire l'influence que la structure de l'intersection (routes, bancs, lampadaires, obstacles en tous genre...) a sur le chemin des agents. Des modèles plus complexes ont été proposés pour prendre en compte ces interactions lors de la prédiction de la future trajectoire d'un agent.

Parmi ces modèles, on retrouve les mécanismes d'attention. Directement importés du domaine du traitement du langage naturel, ils permettent d'entraîner un réseau de neurones à associer automatiquement à chaque élément d'un ensemble une pertinence relative aux autres éléments en se basant sur le contexte de prédiction. Dans le cadre de la traduction de langage, les éléments sont les mots de la phrase à traduire et le contexte de prédiction les mots déjà traduits. Pour la tâche de prédiction, ces modèles ont été transposés de deux manières différentes. Pour prendre en compte les interactions entre un agent et l'intersection qu'il parcourt, on utilise un mécanisme d'attention visuelle qui permet étant donnée l'image d'une scène en vue de dessus, de juger de la pertinence de chaque partie de l'image pour prédire la future position de l'agent. Pour prendre en compte les interactions d'un agent avec les autres agents, on utilise un mécanisme de *soft-attention* permettant de juger de la pertinence de chacun des agents présents dans la scène pour prédire la future position de l'agent. Ces mécanismes d'attention reposent sur l'architecture encodeur/décodeur utilisant des réseaux de neurones récurrents comme modèle de base. Les réseaux de neurones récurrents sont d'ailleurs en général utilisés dans la majorité des études de prédiction de futures

trajectoires en apprentissage profond. Cependant, un petit nombre d'études ont montré que les réseaux de neurones convolutifs étaient capables d'obtenir de meilleures performances que les réseaux de neurones récurrents pour des approches naïves. En transposant directement les mécanismes d'attention du domaine du traitement du langage naturel au domaine de prédiction de trajectoire, aucune interrogation n'est portée sur la sémantique de ces modèles. Ils imposent notamment l'utilisation de l'architecture récurrente encodeur/décodeur, obligeant à prédire séquentiellement les futures positions et à recalculer le module d'attention pour prédire chacune des futures positions. Dans ce travail, nous faisons l'hypothèse que les mécanismes d'attention ainsi proposés apportent une information redondante augmentant inutilement leur temps de prédiction. On propose d'adapter les mécanismes d'attention visuelle et de *soft-attention* afin de pouvoir les utiliser avec des réseaux de neurones convolutifs. On fait l'hypothèse que ces nouvelles architectures permettront de réduire drastiquement le temps de prédiction tout en conservant une qualité de prédiction équivalente.

Néanmoins, il n'est pas facile de définir à quoi correspond une qualité de prédiction équivalente. Deux métriques font consensus pour évaluer la qualité des trajectoires prédites par un modèle. D'abord l'erreur moyenne de déplacement qui permet de mesurer la moyenne de la distance euclidienne entre chaque paire de point de la trajectoire prédite et de la trajectoire réelle. Ensuite, l'erreur finale de déplacement mesurant la distance euclidienne entre le dernier point de la trajectoire prédite et celui de la trajectoire réelle. Certains travaux ont montrés que majoritairement, les modèles naïfs obtiennent de meilleurs résultats que des modèles utilisant les interactions d'un agent avec son environnement selon ces deux métriques. D'autres études montrent que cet effet tend à diminuer quand la complexité et la variété des interactions proposées dans un ensemble de données augmente. On propose de mener notre étude sur le *Stanford Drone Dataset*, un ensemble de données proposant une variété et un nombre d'interactions bien plus important que les ensembles de données généralement utilisés. Pour compléter les modalités d'évaluation nous proposons d'utiliser trois autres métriques (dont deux nouvelles) permettant de juger plus précisément de différentes qualités des modèles appris. Une métrique jugeant de la capacité des modèles à utiliser les interactions des agents entre eux, une jugeant de la capacité des modèles à utiliser les interactions des agents avec la scène dans laquelle ils évoluent et une métrique jugeant du réalisme cinétique des trajectoires prédites. On fait l'hypothèse que ces trois métriques supplémentaires nous permettront une meilleure comparaison des différents mécanismes d'attention.

On montre que les modèles utilisant l'attention visuelle pour tirer partie de l'information spatiale ne permettent pas des gains de performance sur l'ensemble des métriques utilisées dans cette étude. De plus, la métrique proposée pour la prise en compte des interactions d'un agent avec la scène ne permet pas de mettre en évidence un quelconque impact de ces

modèles sur les interactions d'un agent avec son environnement. Ce résultat est partiellement en contradiction avec ceux annoncés dans les études faisant état d'un impact de ces modèles sur la qualité de prédiction pour les erreurs de déplacement.

Les modèles utilisant les mécanismes de *soft-attention* pour prendre en compte les interactions entre agents montrent par contre un impact significatif sur la réduction des collisions entre les trajectoires prédites des agents. L'impact pour le modèle récurrent issu de la littérature et le modèle convolutif proposé dans cette étude sont équivalents. Par ailleurs, le modèle convolutif est au moins aussi bon voire meilleur que son équivalent récurrent pour les erreurs de déplacement des positions prédites ainsi que la qualité des distributions de vitesses et d'accélérations apprises. Cela confirme en partie l'idée que l'information apportée par le module d'attention dans le paradigme récurrent est redondante. L'intérêt de l'approche convolutive est d'autant plus fort qu'elle permet une division du temps de prédiction par sept. On a donc proposé un modèle prenant en compte les interactions sociales plus complet et plus rapide que le modèle déjà existant. La comparaison des modèles attentifs a été permise par les métriques proposées sans lesquelles il aurait été difficile d'évaluer les performances relatives des modèles sur différents aspects. Si elles sont perfectibles, elles constituent un premier pas vers une évaluation plus complète des modèles de prédiction de trajectoire.

ABSTRACT

This work addresses the problem of trajectory prediction of agents of various types such as pedestrian and cyclists interacting with each other within scenes. Predicting agents future paths can be useful in the field of autonomous driving for example. The recent and fast development of Deep Learning algorithms showing groundbreaking performance in fields such as Computer Vision and Natural Language Processing led it to be used for predicting agent future paths. Naïve models, e.g. models that use only past motion in order to predict the future one, have been developed. These models don't make use of the agent environment. The environment is mainly composed of a social part regrouping the interactions between agents, and a spatial part, regrouping the interactions between an agent and the scene in which it moves. More advanced models were built to make use of the environment, based on the idea that the future motion of an agent isn't independent of its surrounding environment. Amongst those models, attention mechanisms stand out. Taken from Natural Language Processing, they allow a neural network to automatically select relevant information from a set of element based on some prediction context. For instance, for language translation, the elements might be the words of the sentence to be translated and the prediction context the already translated words. In the field of future path prediction, attention mechanisms were used in two different ways. A spatial attention mechanism, making use of the Visual Attention model which allows the model to select the relevant parts of an image of the scene, accounting for spatial context, in the next position prediction. And a soft-attention mechanism to select which agents are relevant in predicting one agent next position, accounting for social context. Those attention mechanisms are based on the sequence-to-sequence or (encoder/decoder architecture) using Recurrent Neural Networks as base component. Recurrent-based approaches are mainly used for trajectory prediction. Some studies showed that Convolutional Neural Networks could be used as well, showing better performances than their recurrent counterpart for naïve approaches. By directly transposing the attention mechanisms from Natural Language Processing to trajectory prediction, no interrogation is carried on the semantics of these models. In particular, they require the use of the sequence-to-sequence architecture, making it necessary to predict the future positions sequentially and to recalculate the attention module in order to predict each of the agent next positions. In this work, we make the hypothesis that the attention mechanisms, thus proposed bring redundant information unnecessarily increasing their prediction time. We want to adapt those mechanisms in order to be able to use them with convolutional neural networks. We think that these new architectures will drastically reduce the prediction time while maintaining the prediction quality.

Nevertheless, defining what is a good prediction quality is not an easy task. In the field of trajectory prediction, two criteria are commonly used to evaluate the prediction quality. First, the average displacement error that makes it possible to measure the average of the Euclidean distance between each pair of points of the predicted trajectory and the ground truth trajectory. Then, the final displacement error being the Euclidean distance between the last point of the predicted trajectory and that of the ground truth trajectory.

Some studies have shown that, for the most part, naive models obtain better results than models that use one's agent environment according to these two criteria. Other studies show that this effect tends to decrease when the complexity and variety of interactions in a dataset increases. Since most datasets used in the field are poor in this matter, we use the *Stanford Drone Dataset* which contains a greater number of agents, scenes and interactions. To improve the evaluation methodology, we use three other metrics (two of them new) to account for different qualities of the learned models. A social criterion that evaluates the ability of models to use agent interactions with each other, a spatial criterion to evaluate the ability of a model to use the interactions between an agent and its spatial environment and a criterion that evaluates the kinetic realism of predicted trajectories. We make the hypothesis that these three additional criteria will allow a better comparison of different attention mechanisms.

We show that models using visual attention to take advantage of spatial information do not show any improvement on the social criterion. This result is inconsistent with those reported in past studies, stating that such models have an impact on the quality of prediction.

Social attention models show a significant impact on the reduction of the collisions proportion between agents trajectories that they predicted. The impact of both models on that matter is very similar quantitatively. Moreover, the convolutional model is better in every criteria than its recurrent counterpart, that is to say for the errors of displacement of the predicted positions as well as the quality of the learned distributions of velocities and accelerations. This partly confirms the idea that the information provided by the attention module in the recurrent paradigm is redundant. The advantage of the convolutive approach is all the stronger as it allows a division of the prediction time by seven. We have therefore proposed a model that takes into account social interactions that is more complete and faster than existing one. The proposed evaluation criterias made it possible to evaluate the attention mechanisms in a more comprehensive way. Those criteria while perfectible are a first step toward a new way of evaluating models for trajectory prediction.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
DBN	Dynamic Bayesian Network
LSTM	Long Short Term Memory
MLP	Multi Layer Perceptron
GAN	Generative Adversarial Network
SVM	Support Vector Machine
LCSS	Longest Common Subsequence
DTW	Dynamic Time Warping
HMM	Hidden Markov Model
CVAE	Conditional Variational Autoencoder

LISTE DES ANNEXES

CHAPITRE 1 INTRODUCTION

D'après un rapport du gouvernement de Transport Canada de 2017, les accidents de la route en milieu urbain ont causé 746 morts et 81,938 blessés. C'est dans les carrefours que l'on trouve 30% des blessés et 40% des morts. Cette statistique est d'autant plus importante que les intersections constituent une faible proportion de l'ensemble du réseau routier. Elle met donc en évidence l'importance de s'intéresser plus spécifiquement aux intersections. Ces dernières sont caractérisées par des structures spatiales complexes ainsi que des interactions multiples entre usagers de la route de natures variées comme les piétons, les automobilistes ou encore les cyclistes. Anticiper ou détecter des situations anormales aux intersections est, de fait, un problème important à considérer. Dans ce contexte, prédire les trajectoires futures de tous les usagers présents dans une intersection en temps réel serait crucial. Plus généralement, la capacité d'anticiper le déroulement d'une scène de trafic possède un intérêt dans le cadre des voitures autonomes, de la conduite assistée par ordinateur, de la gestion et la surveillance automatisée du trafic routier, des systèmes d'aide à la conduite ainsi que de la meilleure compréhension des dynamiques prenant place aux intersections, permettant de concevoir des modèles de simulation de ces phénomènes.

Pour mener à bien la tâche de prédiction, il est nécessaire de pouvoir observer le parcours des agents au sein de l'intersection. À ces fins, on peut utiliser les caméras de vidéo. Peu chères depuis les récentes améliorations technologiques, elles pullulent dans les métropoles et plus particulièrement aux intersections. On en recense plus de 500 sur la seule île de Montréal. On peut aussi placer nous-mêmes nos propres caméras. Ces caméras fournissent les données vidéo en temps réel du déroulement du trafic. De ces données, on peut extraire l'historique du déplacement des usagers de la route à l'aide d'algorithmes de détection et de suivi. On dispose alors pour une intersection de l'ensemble des trajectoires des usagers l'ayant parcouru pendant une période de temps définie. On cherche à développer un modèle de prédiction de trajectoires sur une période de temps donnée, c'est-à-dire un modèle nous permettant à partir de l'observation du déplacement passé d'un usager, de prédire son déplacement futur. Fort des données acquises par les caméras, on peut envisager d'utiliser une approche d'apprentissage machine basée sur les données. On pourrait définir manuellement les caractéristiques des trajectoires jugées pertinentes pour développer ce modèle. Cependant, les intersections constituent des scènes contraintes spatialement, théâtres d'interactions complexes entre de nombreux usagers différents. Prédire les futures déplacements est un problème non trivial, au même titre que de définir manuellement les caractéristiques à considérer pour mener à bien cette tâche de prédiction.

1.1 Définitions et concepts de base

Dans ce travail, on désigne par **agent** ou **usager de la route** n'importe quel individu (ou personne) se déplaçant sur la route ou dans les espaces publics adjacents, peu importe son moyen de locomotion. Un agent peut être, par exemple un piéton, un cycliste ou encore un automobiliste. On désigne par **trajectoire**, une série temporelle à deux dimensions exprimant la position d'un agent dans un repère cartésien orthonormé en fonction du temps. Si dans la réalité une trajectoire est de nature continue, en pratique, elle est mesurée par une série discrète de points séparés par un intervalle de temps supposé constant. Le nombre de points par seconde est appelé **taux d'échantillonnage**. La trajectoire d'un agent prend place au sein d'une **scène**, celle-ci est une entité limitée à la fois dans l'espace et le temps, caractérisée entre autre par la complexité de sa structure spatiale, ainsi que la variété en type et en nombre des agents qui la parcourent. Elle dispose de caractéristiques structurelles spécifiques comme par exemple la disposition des trottoirs, le nombre de routes entrantes et sortantes, ou sa signalisation. Le **problème de prédiction** est approché de la manière suivante : on observe la trajectoire d'un agent pendant une période de temps définie (**période ou horizon d'observation**) appelée **trajectoire observée**. On cherche ensuite à prédire la trajectoire de l'agent pendant une période de temps immédiatement consécutive (**période ou horizon de prédiction**) appelée **trajectoire prédite**. **L'historique des trajectoires d'une scène** fait référence à l'ensemble des trajectoires des agents ayant traversé la scène pendant la période de temps considérée pour cette scène. Le problème de prédiction (figure 1.1) dans ce travail peut-être considéré comme une **prédiction à moyen terme** et est défini de la manière suivante : on observe un agent pendant 3.2 secondes et on prédit son déplacement pour les 4.8 secondes suivantes.

La **structure spatiale de la scène** renvoie à son aménagement. Cela renvoie entre autre à l'agencement des routes, des trottoirs ou des principaux axes de circulation ainsi qu'au mobilier urbain tel que les bancs et les lampadaires. En somme, tout élément statique de la scène pouvant avoir une influence sur le déplacement des individus en son sein. Dans une scène, on parle de **patron de déplacement** pour désigner l'ensemble des tendances principales de déplacement utilisées par les agents la parcourant. Les chemins les plus empruntés par exemple. On dit qu'une scène dispose d'une **structure spatiale complexe**, si cette dernière impose un grand nombre de **patrons de déplacement**. On parlera de **structure spatiale simple** si la scène n'impose qu'un nombre restreint de **patrons de déplacement**, c'est-à-dire que tous les agents tendent à emprunter le même chemin.

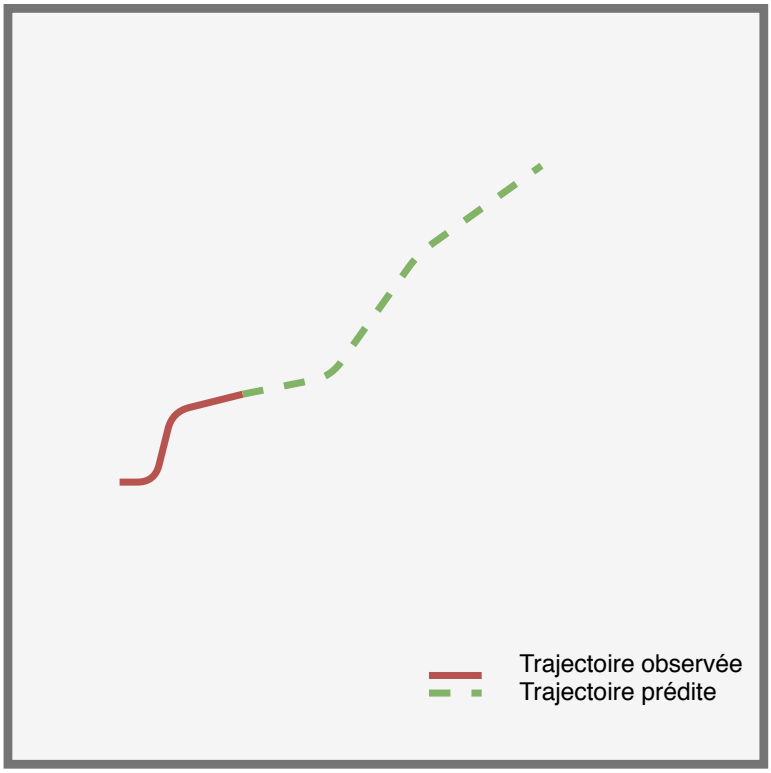


Figure 1.1 Représentation du problème de prédiction de trajectoire

Dans le problème de prédiction, l'agent dont on veut prédire la future trajectoire est appelé **agent principal**. Les autres agents dont on peut observer la trajectoire en parallèle de celle de l'agent principal sont appelés **agents voisins** de l'agent principal. On désigne par **interaction sociale** ou **interaction agent/agent**, les interactions entre l'agent principal et chacun de ses voisins. Le **contexte social** de l'agent principal renvoie à l'ensemble de ces interactions. On désigne par **interaction spatiale** ou **interaction agent/espace**, les interactions entre l'agent principal et l'ensemble des éléments de la structure spatiale de la scène qu'il parcourt. Le **contexte spatial** de l'agent principal renvoie à l'ensemble de ces interactions. On parle d'**attributs** ou de **caractéristiques** indifféremment pour désigner un ensemble de valeurs permettant de définir un objet. Les **modèles ou approches de prédiction naïfs** désignent ceux qui effectuent les prédictions uniquement à partir de la trajectoire observée, sans prendre en compte ni le contexte social, ni le contexte spatial. Par opposition, un **modèle ou une approche est dit complexe** si il utilise pour faire ses prédictions, le contexte social ou spatial, ou les deux. On parlera de **modèle attentif** si le modèle d'apprentissage profond utilise un mécanisme d'attention. Pour un modèle d'apprentissage profond, on dira que l'on utilise le **paradigme récurrent** si son architecture est principalement axée autour des réseaux de neurones récurrents (RNN). Pour un modèle d'apprentissage profond,

on dira que l'on utilise le **paradigme convolutif** si son architecture est principalement axée autour des réseaux de neurones convolutifs (*CNN*).

Figure 1.2 Une intersection spatialement complexe et un sous-ensemble de ses patrons de déplacement

Sur la figure 1.2, on peut observer un exemple de scène dont la structure spatiale est relativement complexe. Quelques exemples de modalités de déplacement possibles au sein de la scène sont représentées par les flèches de couleur.

1.2 Problématique

Depuis la fin des années 90, différentes approches ont été proposées pour mener à bien la tâche de prédiction de la trajectoire d'un agent évoluant dans une scène. Des méthodes physiques basées sur la dynamique des véhicules (Tran et al. [7]), performantes à très court-terme (< 1 s) mais incapables de prendre en compte les interactions d'un agent avec son environnement. Des méthodes basées sur l'identification des différents patrons de mouvements possibles au sein d'une scène (Morris et al. [8]) prenant ainsi en compte l'influence de la structure spatiale de la scène mais négligeant les interactions entre les différents agents

présents dans une scène. Des approches utilisant les Réseaux Bayésiens Dynamiques (DBN) (Ballan et al. [9]) proposent de prendre en compte de la manière dont un agent traverse une scène en fonction de son type en représentant de manière probabiliste le passage d'un agent d'une partie à une autre de la scène étant donné des critères comme son type et les caractéristiques de traversabilité de la scène obtenues à partir de l'historique des trajectoires de la scène permettant de bien utiliser le contexte spatial lors de la prédiction.

Ces méthodes présentent trois écueils principaux. D'abord, elles nécessitent de définir manuellement les attributs jugés pertinents pour la tâche de prédiction. Ensuite, les modèles permettant de représenter le contexte spatial de prédiction ne peuvent pas être généralisés à de nouvelles scènes lorsqu'on ne ne dispose pas de l'historique des trajectoires et ne peuvent pas être entraînés conjointement sur des trajectoires provenant de scènes différentes. Enfin, aucun modèle ne propose de solution pour prendre en compte les interactions entre agents lors d'une prédiction. Fort de son succès dans d'autres domaines comme la vision par ordinateur avec les *CNN* de Lecun et al. [10] ou le traitement du langage naturel avec les *RNN*, l'apprentissage profond est utilisé dans le domaine de la prédiction de trajectoire pour répondre à ces trois préoccupations. De par leur prévalence pour les problèmes de traitement de séquences, les *RNN* sont majoritairement utilisés dans le domaine de l'apprentissage profond appliqué à la prédiction de trajectoires. Des approches permettant de prendre en compte les interactions sociales et spatiales sont proposées par exemple par Lee et al. [11]. Dans ce travail, nous nous limiterons exclusivement à l'étude des approches utilisant l'apprentissage profond étant donnée leur capacité supposée à généraliser la connaissance à l'échelle de plusieurs de scènes différentes.

Parmi les approches d'apprentissage profond, un nombre restreint d'entre elles utilisent les mécanismes d'attention. Ces mécanismes permettent d'entraîner un modèle à associer automatiquement un poids aux divers éléments composant son environnement. Par exemple, dans le cadre des interactions sociales, admettons qu'un agent soit entouré de n agents voisins et que l'on souhaite prédire sa future trajectoire en tenant compte du contexte social. Le mécanisme d'attention permettrait d'associer à chaque agent voisin un poids correspondant à l'importance à associer à chacun d'entre eux pour réaliser la prédiction. Ces mécanismes présenteraient deux intérêts majeurs. D'abord ils seraient intéressants pour la sélection automatique d'attributs pertinents pour la tâche de prédiction. Ensuite ils permettraient une meilleure interprétabilité des décisions d'un modèle en connaissant les poids associés par le modèle à chaque objet de l'environnement lors d'une prédiction. Si les mécanismes d'attention ont été beaucoup étudiés dans d'autres domaines de l'apprentissage profond, ce n'est pas le cas pour celui de la prédiction de trajectoires. Les études à ce sujet utilisent essentiellement un mécanisme d'attention transposé depuis le domaine du traitement du langage naturel. Ce

mécanisme est utilisé conjointement aux *RNN*. On remarque deux choses :

- Le paradigme récurrent majoritairement utilisé dans le domaine ne donne pas de meilleurs résultats que le paradigme convolutif (Nikhil et al [5]), or les calculs dans les *CNN* sont facilement parallélisables contrairement aux *RNN*. Ils sont de fait plus rapides pour effectuer une prédiction. La rapidité d'exécution peut faire une différence importante si on veut considérer l'ensemble des interactions sociales pour des scènes peuplées (plusieurs dizaines d'agents) et effectuer des prédictions dans un temps raisonnable.
- La sémantique portée par le mécanisme d'attention utilisé est potentiellement redondante et par conséquent inutilement coûteuse en temps de calcul.

Dans ce mémoire, on étudie la pertinence des mécanismes d'attentions proposés dans le paradigme récurrent. On propose pour pousser plus loin cette étude, d'adapter les mécanismes d'attention au paradigme convolutif. Le changement de paradigme entraîne un changement de la sémantique portée par le module d'attention. On cherche à savoir si les modules d'attentions transposés dans le paradigme convolutif peuvent permettre de réaliser des prédictions de qualité équivalente. On étudie aussi l'impact du changement de paradigme sur les temps nécessaires pour effectuer une prédiction.

Si l'on veut étudier l'impact de l'utilisation des interactions d'un agent avec son environnement social et spatial sur les performances de prédiction, il est nécessaire de disposer de métriques adéquates pour juger de l'impact de ces informations supplémentaires. Il est aussi nécessaire de disposer d'un ensemble de données suffisamment complexe. C'est-à-dire un ensemble de données proposant des scènes imposant des patrons de mouvements variés ainsi qu'une densité importante d'agents. L'absence de ces deux critères réduit l'impact de l'environnement sur les trajectoires des agents, empêchant d'évaluer correctement l'apport d'un modèle prenant en compte l'environnement par rapport à un modèle naïf (Hug et al. [12]). La plupart des études du domaine utilisent des ensembles de données, des métriques ainsi que des protocoles de validation divers. Cette variété importante rend difficile la comparaison entre les différentes études et permet difficilement de se faire une idée sur la pertinence des méthodes proposées. Les ensembles de données et métriques utilisés sont potentiellement inadaptés pour évaluer de manière satisfaisante l'impact des différentes approches complexes utilisées. Ces considérations nous conduisent à définir nos objectifs de recherche.

1.3 Objectifs de recherche

Le but principal de ce mémoire est l'étude, dans le cadre de l'apprentissage profond pour la prédiction de trajectoires, des mécanismes d'attention. L'objectif secondaire est de

proposer un nouveau cadre d'évaluation pour la tâche de prédiction de trajectoire. Dans un premier temps, on propose un nouveau cadre d'évaluation inspiré de celui du challenge *trajnet* de Sadeghian et al. [13]. On utilisera un ensemble de données riche en patrons de déplacement et interactions sociales, composé de scènes ayant une structure spatiale complexe. En plus des deux métriques proposés dans ce challenge, et de la métrique pour prendre en compte les interactions sociales proposée par Sadeghian et al. [6], on proposera deux nouvelles métriques :

— Une métrique pour mieux évaluer l'impact de la prise en compte des interactions spatiales dans les modèles.

— Une métrique pour évaluer le réalisme de la dynamique des trajectoires prédites par nos modèles.

On mesurera en outre pour chaque modèle, le temps de prédiction d'une trajectoire. Dans un deuxième temps, on évalue l'intérêt des modèles utilisant l'attention, déjà proposés dans la littérature, se reposant sur le paradigme des *RNN*. On propose ensuite des modèles d'attention adaptés au paradigme convolutif, pour évaluer l'impact du changement de paradigme tant sur la qualité des résultats que sur la rapidité de prédiction. On proposera séparément des modèles utilisant le contexte social et des modèles utilisant le contexte spatial. Dans un troisième temps, on essaie de prédire simultanément les trajectoires de tous les agents impliqués dans une même interaction pour mieux prendre en compte les interactions spatiales. Par conséquent, ce travail étudie les hypothèses suivantes :

1. Les mécanismes d'attention fonctionnent dans le paradigme convolutif aussi bien que dans le paradigme récurrent.
2. Les modèles avec attention convolutifs améliorent la rapidité de prédiction pour une trajectoire par rapport aux modèles avec attention récurrents.
3. Les nouvelles métriques permettent une meilleure évaluation de la capacité des modèles à prédire des trajectoires réalistes.
4. Prédire conjointement les futures trajectoires de tous les agents présents dans une scène améliore la prise en compte des interactions sociales.
5. Prédire conjointement les futures trajectoires de tous les agents présents dans une scène améliore la temps de prédiction d'une trajectoire.

1.4 Contributions

Les contributions de ce mémoire sont les suivantes :

1. Un modèle convolutif avec attention prenant en compte interactions sociales sept fois plus rapide et plus performant qualitativement que son pendant récurrent pour tous

les critères d'évaluation.

2. Un modèle convolutif avec attention prenant en compte interactions spatiales neuf fois plus rapide et aussi performant qualitativement que son pendant récurrent.
3. Un nouveau cadre d'évaluation des résultats prenant en compte des interactions plus complexes et plus nombreuses d'un agent avec son environnement.
4. Deux métriques prenant en compte les interactions agent/espace, et la dynamique des trajectoires pour évaluer qualitativement la capacité prédictive des modèles.
5. Un bilan au regard de ce nouveau cadre d'évaluation de l'impact des mécanismes d'attention déjà proposés dans la littérature.

1.5 Plan du mémoire

Le second chapitre de ce mémoire fait une revue de littérature en trois parties : la première se concentre sur les domaines plus généraux de l'apprentissage profond nécessaire à notre étude, la deuxième fait l'historique du domaine de la prédiction de trajectoires, la troisième revient plus en détail sur certaines approches de l'apprentissage profond pour la prédiction de trajectoire relatives à notre étude. Le troisième chapitre décrit la méthodologie employée. Le quatrième chapitre décrit les expériences effectuées pour tester les hypothèses de recherche. Le cinquième énonce les résultats obtenus. Enfin, le dernier chapitre analyse les résultats et conclut.

CHAPITRE 2 REVUE DE LITTÉRATURE

2.1 Connaissances préliminaires en apprentissage profond

Dans cette première partie de la revue de littérature, nous revenons sur les éléments théoriques en apprentissage profond nécessaires à une bonne compréhension de l'étude portée par ce mémoire.

2.1.1 Définition du problème de d'appairage d'une séquence à une autre

Le problème se définit de la manière suivante. On dispose d'une séquence S_x correspondant à une liste ordonnée d'observations (toutes du même type) à des instants réguliers. Chaque observation est caractérisée par un vecteur d'attributs (par exemple une trajectoire est une liste de points caractérisés par leurs coordonnées dans l'espace). On veut prédire une séquence S_y. Cette séquence peut être la même que S_x comme pour les auto-encodeurs (Hinton et al. [14]), ou différente de S_x comme dans le cas de la traduction de langage (Sutskever et al. [15]). Entraîner un modèle à prédire S_y étant donné S_x peut être visualisé comme apprendre au modèle à modéliser la probabilité conditionnelle :

$$P(S_y|S_x) \tag{2.1}$$

Si en plus de S_x, on possède une information complémentaire c, et que l'on utilise cette information pour la tâche de prédiction, on parlera dans ce mémoire de *conditionnement*. On dira que l'on conditionne sur c. Entraîner un modèle à prédire S_y étant donné S_x et c peut être alors visualisé comme apprendre au modèle à modéliser la probabilité conditionnelle :

$$P(S_y|S_x, c) \tag{2.2}$$

2.1.2 Réseaux de Neurones Récurrents

Les réseaux de neurones classiques (ou *Multi-Layer Perceptron (MLP)*) se montrent efficaces sur une large variété de tâches d'apprentissage machine. Cependant, ils souffrent de certaines limitations inhérentes à leur architecture. Etant donné que le nombre d'entrées d'un *MLP* est fixe, il ne peut pas être entraîné sur des exemples d'entraînement de tailles variables. Ainsi, les *MLP* n'ont pas initialement été conçus pour traiter des problèmes de séquences ou de séries temporelles. Afin de pallier ce problème, une architecture spécialement adaptée a

été proposée : les *RNN*.

Les *RNN*, contrairement aux *MLP*, sont capables de considérer des entrées séquentielles, mais aussi de modéliser une dépendance temporelle entre les divers points qui composent cette séquence. Ainsi, quand un *RNN* effectue une prédiction pour un élément d'une séquence au temps t, il prend en compte les prédictions qu'il a réalisées pour les éléments antérieurs de la séquence. D'un point de vue pratique, cela nécessite d'introduire un mécanisme de mémorisation, de telle sorte que la prédiction à l'instant t utilise à la fois la mémoire à l'instant *t-1* et l'élément de la séquence d'entrée à l'instant t. On définit un état caché h_{t-1} comme la mémoire du réseau à l'instant *t-1*, x_t l'entrée du *RNN* à l'instant t et deux matrices de poids U et V. L'état caché à l'instant t est défini de la manière suivante :

$$h_t = \phi(Vh_{t-1} + Ux_t), \tag{2.3}$$

Où ϕ est la fonction sigmoïde ou tangente hyperbolique. On peut remarquer qu'étant donné le caractère récurrent de sa définition que h_t dépend de tous les h_i antérieurs, avec $i < t$. Ce mécanisme est différentiable, car composé uniquement d'opérations différentiables. Le *RNN* peut ainsi être entraîné par rétro-propagation. Le *RNN* peut être vu comme un *MLP* qui devient récurrent au fur et à mesure qu'il se déploie selon la dimension temporelle (figure 2.1).

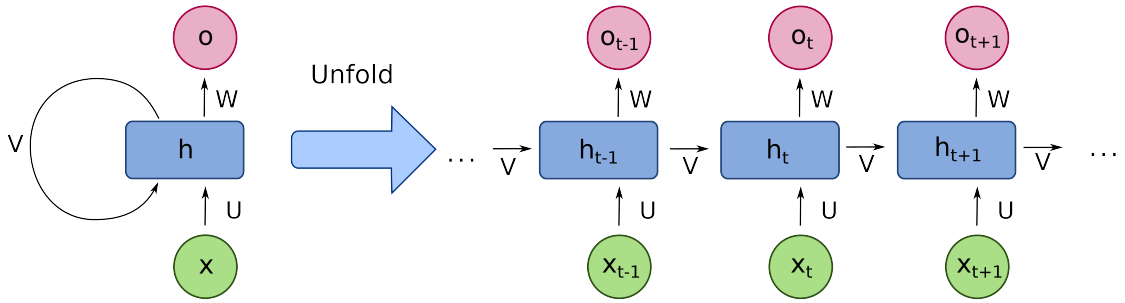


Figure 2.1 Représentation du déploiement d'un *RNN* dans le temps repris de Deloche [1]

Le caractère récurrent de cette approche rend la rétro-propagation instable pour les longues séquences. La rétro-propagation consiste essentiellement en une suite de produits de gradient de l'activation d'une couche cachée par les poids qui la composent. Si les gradients sont inférieurs à un, alors au fur-et-à-mesure de leurs multiplications consécutives, ils tendent

à devenir très petit et à disparaitre : c'est le problème de la disparition de gradient. Hochreiter et al. [16] proposent en 1997 le *Long Short-Term Memory* (*LSTM*) pour résoudre ce problème et traiter des séquences de longueurs supérieures à mille éléments.

Le *LSTM* se base sur la cellule représentée dans la figure 2.2 et introduit deux éléments principaux. Premièrement, en lieu et place d'avoir une simple couche cachée complètement connectée, il en possède quatre qui interagissent entre elles au sein d'une cellule. Deuxièmement, en plus de l'état caché propre au *RNN*, le *LSTM* dispose d'un mécanisme de mémoire supplémentaire appelé état de la cellule C_t.

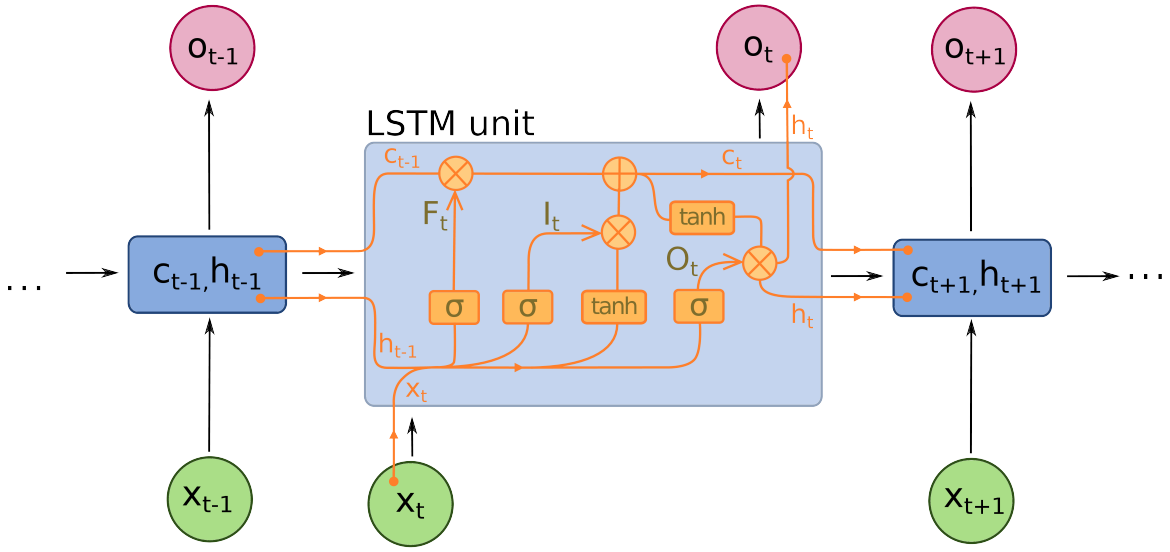


Figure 2.2 Cellule du *LSTM* et son déploiement dans le temps repris de Deloche [2]

L'état de la cellule représente la mémoire à long-terme de la cellule (C_t ne subit que des transformations mineures : somme et produit élément à élément) par opposition à l'état caché qui représente la mémoire à plus court terme (h_t subit des transformations plus complexes, fonctions sigmoïde et tangente). Le mécanisme est divisé en trois parties principales :

— La porte d'oubli F_t : étant donné l'entrée x_t et l'état caché h_{t-1}, permet d'oublier une partie de l'information mémorisée dans l'état de la cellule C_{t-1}.

— La porte d'entrée I_t : étant donné l'entrée x_t et l'état caché h_{t-1}, permet d'ajouter une partie de l'information en entrée dans l'état de la cellule C_{t-1}.

— La porte de sortie O_t : applique une fonction non linéaire sur l'entrée x_t.

La sortie du réseau o_t équivalent à l'état caché en sortie h_t : étant donnée l'état de la cellule mis-à-jour, C_t et le résultat de la porte de sortie O_t, permet de filtrer l'information en entrée à transmettre dans l'état caché en sortie h_t. L'état caché et l'état de la cellule sont transmis

de cellule en cellule au cours du temps. En pratique, la mise-à-jour de l'état caché et de l'état de la cellule sont réalisés de la manière suivante :

$$
\begin{aligned}
I_t &= \sigma(U^i x_t + W^i h_{t-1}) \\
F_t &= \sigma(U^f x_t + W^f h_{t-1}) \\
O_t &= \sigma(U^o x_t + W^o h_{t-1}) \\
C_t &= \sigma(F_t \cdot C_{t-1} + I_t \cdot tanh(U^g x_t + W^g h_{t-1})) \\
h_t &= o_t = \tanh(C_t) \cdot O_t
\end{aligned}
$$

Avec σ la fonction sigmoïde, $\cdot$ le produit matriciel terme-à-terme et les matrices U^j et W^k correspondant à des matrices de poids.

2.1.3 Prédiction de séquences avec les LSTM

Les *LSTM* peuvent être utilisés pour prédire une séquence en sortie à partir d'une séquence en entrée. Deux méthodes sont couramment utilisées :

— Si la longueur de la séquence à prédire est fixe, on utilise la variante *LSTM-MLP*.

— Sinon, on utilise la variante *LSTM* encodeur-décodeur.

Dans *LSTM-MLP*, la séquence est traitée par le *LSTM*. On obtient pour chaque objet de la séquence, un état caché lui correspondant. Le dernier état caché h_t est supposé contenir l'information condensée de tous les objets de la séquence. Cette représentation est de taille fixe. Elle est ensuite fournie en entrée d'un *MLP* dont la couche de sortie aura pour taille le produit de la longueur attendue de la séquence prédite et du nombre d'attributs par objet.

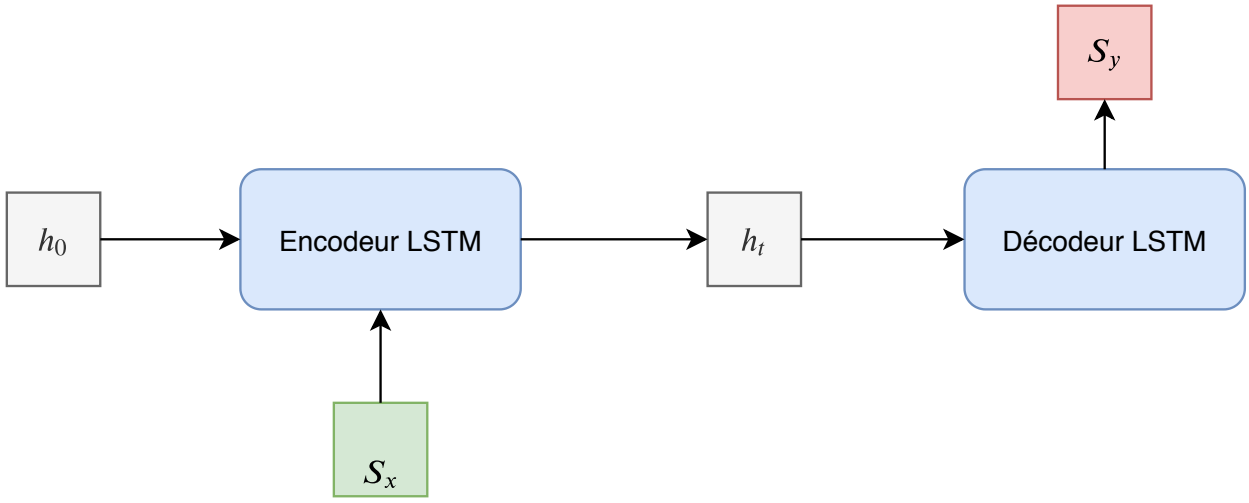


Figure 2.3 LSTM encodeur décodeur

LSTM encodeur-décodeur (figure 2.3), introduit par Sutskever et al. [15], fonctionne de la manière suivante. Comme dans *LSTM-MLP*, la séquence en entrée S_x est encodée sous la forme du dernier état caché h_t de l'encodeur *LSTM*. Cet état caché est utilisé pour conditionner le décodeur *LSTM* avec le contexte de la séquence en entrée. Le conditionnement est réalisé en initialisant l'état caché du décodeur avec le dernier état caché de l'encodeur. De plus, l'entrée du décodeur peut être utilisée entre autre des quatre manières suivantes :

— Pour chaque prédiction, le décodeur prend en entrée sa prédiction précédente de manière récursive.
— Pour chaque prédiction, le décodeur prend en entrée une information complémentaire permettant de raffiner le contexte de prédiction.
— Pour chaque prédiction, le décodeur ne prend pas d'entrées et les états internes du décodeur sont mis à jour sans prendre l'entrée en compte. Chaque prédiction dépend alors uniquement de l'état caché précédent.
— Pour chaque prédiction, le décodeur prend en entrée sa prédiction précédente et une information complémentaire pour conditionner la prédiction.

2.1.4 Réseaux de neurones convolutifs

Les *CNN* ont été introduits en 1999 par LeCun et al. [17]. Le principe est le suivant, le réseau prends en entrée un tenseur (de trois dimensions dans le cas de la convolution 2D), le plus souvent une image. Les dimensions du tenseur dans le cas d'une image sont : la hauteur, la largeur et la profondeur. Le tenseur subit plusieurs transformations consécutives au sein

du réseau. Elles consistent en une alternance de couches de convolution et de couches de réduction. Au terme de la dernière couche de réduction, les vecteurs d'attributs obtenus sont aplatis dans un vecteur, puis passés dans un *MLP* qui s'occupe de réaliser la prédiction finale.

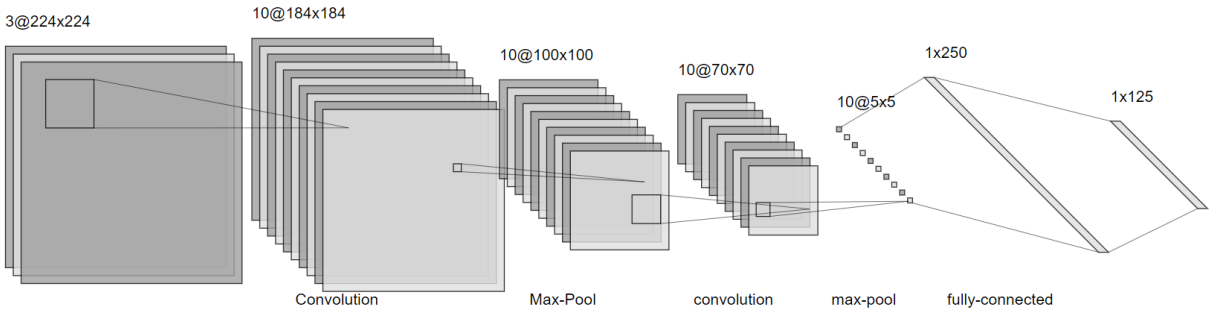


Figure 2.4 Un exemple de réseau de neurone convolutif

Une couche de convolution est définie par plusieurs paramètres :

- Le nombre de filtres de convolution à appliquer sur le tenseur en entrée.
- La hauteur et la largeur des filtres de convolutions.
- L'intervalle de déplacement entre deux applications consécutives d'un filtre sur le tenseur.
- La dimension du rembourrage (nombre de zéros à ajouter de part et d'autre de la hauteur et de la largeur du tenseur). Il est utilisé si la taille des entrées utilisées, par exemple des images, est variable.

L'application de N filtres de convolution, permet d'obtenir N vecteurs d'attributs. La dimension de ces vecteurs d'attributs est régie par les paramètres précédemment mentionnés. Un filtre de convolution 2D est une matrice de valeurs réelles. Les valeurs des filtres de convolution ne sont pas définies manuellement. Elles sont apprises par le réseau pendant la rétro-propagation, afin d'extraire automatiquement les caractéristiques les plus pertinentes du tenseur en entrée. La couche de convolution est suivie par une fonction d'activation de type *Relu* (équation 2.1.4) le plus souvent .

$$Relu(x) = max(0, x) \tag{2.4}$$

La couche de réduction de dimension permet de réduire les dimensions de chaque vecteur d'attributs en les agrégeant par sous-régions spatiales. On peut utiliser plusieurs fonctions

de réduction. Sur la figure 2.4 on utilise une réduction par le maximum (*Max-Pool* sur la figure), c'est-à-dire que l'on remplacera l'ensemble des valeurs d'une région du tenseur par une seule valeur, le maximum.

2.1.5 Réseaux de neurones convolutifs, pré-entraînement et transfert de connaissances

Les réseaux convolutifs se révèlent très efficaces pour le traitement et l'extraction automatique d'attributs des images en raison de l'affinité particulière de leurs opérations pour traiter des données structurées spatialement. Néanmoins, pour entraîner des modèles performants, capables d'extraire des attributs riches et sur des tâches non triviales (classification parmi mille classes d'image comme le challenge *ImageNet* (Russakovsky et al. [18])), il est nécessaire de disposer d'ensembles de données conséquents (plusieurs millions d'images) ainsi que des ressources de calcul idoines.

Malgré cette contrainte, il est possible de tirer parti de modèles complexes déjà entraînés de deux manières :

— On peut utiliser des *CNN* déjà entrainés afin d'extraire automatiquement des attributs à partir des tenseurs d'entrée. Il suffit alors d'ôter la partie *MLP* du *CNN* pour récupérer les vecteurs de caractéristiques extraits par le *CNN*. Le *CNN* peut être utilisé comme module auxiliaire, sans que ses poids ne soient mis à jour pendant la rétro-propagation. Des *CNN* pré-entraînés spécialisés dans l'extraction d'attributs à partir d'images, tels que *VGG-net* proposé par Simonyan et al. [19] sont disponibles en ligne.

— Le transfert de connaissance reprend l'idée précédente, on utilise un modèle pré-entrainé dont on enlève le dernier composant chargé de la prédiction. On remplace ensuite ce composant par un autre, et on ré-entraîne le réseau ainsi obtenu, partiellement ou complètement, sur une tâche plus ciblée que celle de l'entraînement initial du réseau. On parle de raffinement du réseau. On peut par exemple ré-entrainer le réseau sur une tâche similaire avec un ensemble de données plus spécifique ou encore changer de tâche, comme par exemple, passer d'une classification à une segmentation d'image (Long et al. [20]).

2.1.6 Réseaux de neurones convolutifs pour le traitement de séquences

Malgré les associations tacites entre *RNN* avec le traitement des séquences et *CNN* avec le traitement des images, différentes approches récentes ont essayé d'adapter les *CNN* aux problèmes de séquences. L'intérêt principal de cette démarche est explicité par Gehring et

al. [21], les calculs dans les *CNN* peuvent être complètement parallélisés par opposition à ceux des *RNN* et permettent de mieux exploiter les capacités des cartes graphiques. Cette différence est inhérente aux architectures des deux familles de modèles. Dans les *RNN*, le traitement d'un élément de la séquence dépend du traitement des éléments le précédant. Dans les *CNN*, les convolutions sont des opérations indépendantes pouvant être effectuées en parallèle. Gehring et al. [21] obtiennent des meilleures performances dans les tâches de traduction anglais-français et anglais-allemand que le modèle récurrent équivalent, tout en divisant par dix le temps de prédiction.

Bai et al. [3] proposent le *Temporal Convolutional Network (TCN)*, une architecture de réseau convolutif dédiée aux problèmes de séquences censée être équivalente au *LSTM*. Le modèle est testé contre le *LSTM* sur une variété de problèmes dans lesquels ce dernier se spécialise. Les résultats suggèrent que le *TCN* est capable de faire aussi bien voire mieux que le *LSTM* sur la majorité des problèmes, marquant le début de la fin de l'hégémonie des réseaux récurrents pour le traitement des séquences.

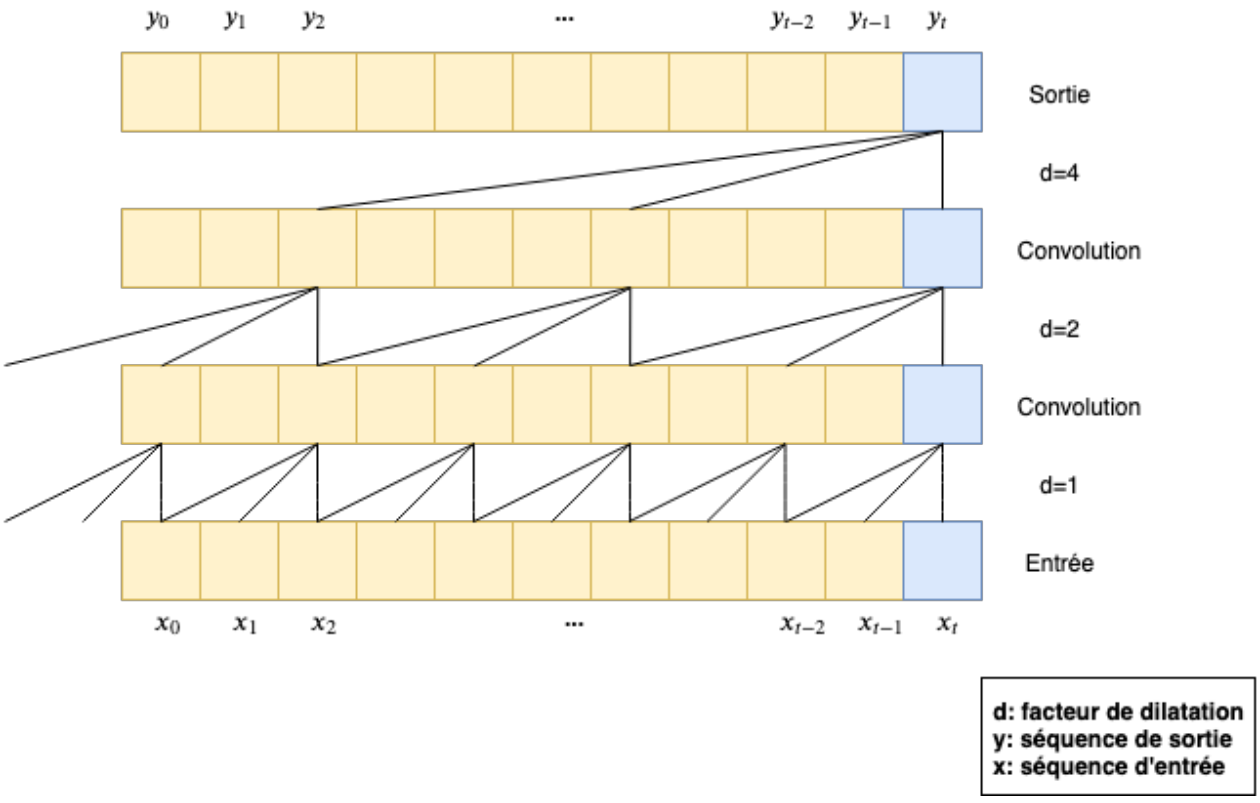


Figure 2.5 Convolution causale dilatée adapté de [3]

Le *TCN*, consiste en une convolution 1D le long de la dimension temporelle de la séquence, avec deux adaptations majeures par rapport au *CNN* classique :

— La convolution causale (figure 2.5) signifie que le vecteur de caractéristiques résultant de l'énième convolution n'utilise que des éléments antérieurs à l'énième objet de la séquence d'entrée. On ajoute autant de zéros que nécessaire (soit la taille du filtre de convolution moins un) au début de la séquence de telle sorte que pour l'énième convolution, le dernier élément considéré par le filtre de convolution soit l'énième élément de la séquence en entrée. Par exemple sur la figure 2.5 on ajoute deux zéros à la séquence en entrée afin que le dernier élément considéré par la première convolution de la première couche soit l'élément x_0.

— La convolution dilatée. Si on applique une simple convolution, le champs réceptif du dernier vecteur d'attributs fourni par le réseau n'est obtenu qu'à partir d'une quantité, linéaire à la profondeur du réseau, des points de la séquence initiale. Cela pose problème pour les séquences très longues relativement à la taille du filtre de convolution, où il faudrait des réseaux très profonds pour avoir le champs réceptif requis. Comme on veut utiliser le dernier vecteur d'attributs de manière similaire au dernier état caché du *LSTM*, il est nécessaire que celui-ci prenne en compte l'ensemble des points de la séquence en entrée. Dans ce but, on applique une dilatation progressive du filtre de convolution. A chaque couche de convolution successive, celui-ci est multiplié par deux, permettant d'obtenir un champs réceptif exponentiel par rapport au nombre de couches du réseau, diminuant ainsi le nombre de couches totales nécessaires. La convolution causale dilatée est représentée par la figure 2.5.

Utiliser des *CNN* pour traiter des problèmes de séquence impose de fixer une longueur maximale pour les séquences en entrée. Ce qui peut être réalisé pour la majorité des problèmes utilisant des données séquentielles.

2.1.7 Mécanismes d'attention

Les mécanismes d'attention ont fait leur première apparition dans le domaine de la traduction de langages et plus généralement dans le domaine du traitement du langage naturel. La majorité des modèles de traduction utilisent l'architecture *LSTM* encodeur/décodeur. Pour rappel, la séquence en entrée (ici une phrase), est passée objet par objet dans l'encodeur *LSTM*. On obtient un état caché par objet présent dans la séquence (ou mot dans la phrase). Le dernier état caché produit est censé, en théorie, grâce au mécanisme de mémorisation du *LSTM*, représenter le sens de l'ensemble de la phrase. En pratique, plus la phrase est longue, moins le dernier état caché est capable de représenter l'ensemble de la phrase et plus il représente la fin de celle-ci. On se rend bien compte que pour traduire une phrase, il

serait plus optimal de se concentrer en alternance sur différents mots de la phrase en entrée. Chacun des états cachés obtenus représente le mieux le dernier objet (ou dernier mot) considéré par le *LSTM*. Le but initial d'un module d'attention est ainsi de choisir étant donné le contexte quel objet de la séquence (quel mot de la phrase) en entrée utiliser pour effectuer la prochaine prédiction. Par exemple, sur la figure 2.6 on traduit une phrase du français à l'anglais. Chaque mot de la phrase en français est passé tour à tour dans l'encodeur *LSTM*. Pour chaque mot en entrée, on obtient un encodage $h_{enc,mot}$. Il est important de se rappeler que cet encodage encode à la fois le mot en question ainsi que tous les mots précédents. L'état caché du décodeur *LSTM* est initialisé avec $h_{enc,mémoire}$ qui encode l'ensemble de la phrase en entrée, ce qui permet de transmettre le contexte de prédiction au décodeur. Le reste de la figure illustre que le décodeur pour prédire le mot "student" va se concentrer principalement sur $h_{enc,étudiant}$ soit l'encodage du mot "étudiant" (et des précédents), qui s'avère être l'encodage le plus pertinent à considérer en l'occurence.

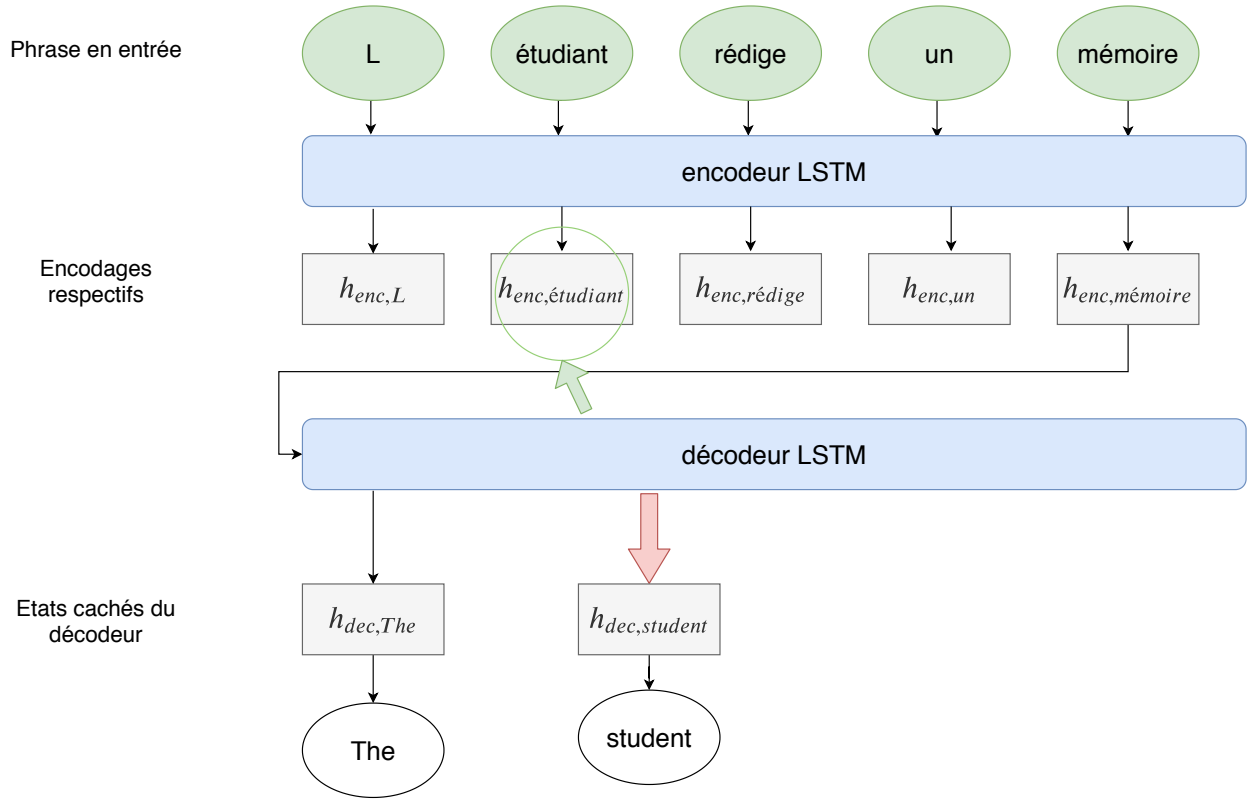


Figure 2.6 Traduction du français à l'anglais en utilisant un LSTM encodeur/décodeur et un mécanisme d'attention

Ainsi pour faire simple un mécanisme d'attention est un mécanisme qui, étant donné un contexte de prédiction (représenté dans l'exemple par l'état caché du décodeur à l'instant précédent $h_{dec,mot}$) va permettre de sélectionner les objets les plus pertinents pour la prédiction courante (ici les mots les mots de la phrase en entrée) en se basant sur un ensemble de représentation de ces objets (ici les encodages des mots de la phrase en entrée $h_{enc,mot}$).

Dans le modèle encodeur/décodeur *LSTM* les prédictions sont effectuées de manière récurrente, et le mécanisme est utilisé pour chaque prédiction. Ce qui change d'une prédiction à l'autre, c'est le contexte de prédiction ou état caché du décodeur). Pour généraliser cette idée, on reprend la terminologie de Vaswani et al [4]. On appelle *valeurs* l'ensemble des objets sur lesquels on applique le mécanisme d'attention et on note V la matrice regroupant l'ensemble des vecteurs d'attributs de ces *valeurs*. On appelle *clés* un ensemble d'éléments. Il y a autant de *clés* que de *valeurs*. Chaque *clé* est appairé à une *valeur*. La majorité du temps, les *clés* sont les mêmes objets que les *valeurs*. Parfois, on peut choisir de projeter les vecteurs d'attributs des *clés* en utilisant un *MLP*. Le choix de projeter ou non dépend essentiellement du mécanisme d'attention que l'on utilise. On note K la matrice regroupant l'ensemble des vecteurs d'attributs de ces *clés*. Enfin on appelle *requête* l'objet représentant le contexte permettant de sélectionner les *valeurs* les plus pertinentes. Il peut y avoir plusieurs *requêtes* simultanément si on calcule plusieurs vecteurs d'attention en parallèle. On note Q la matrice regroupant l'ensemble des vecteurs d'attributs de ces *requêtes*.

Ainsi, en généralisant le concept du module d'attention autour de cette terminologie, un mécanisme d'attention est un mécanisme qui étant donné une *requête* va permettre de sélectionner les *valeurs* les plus pertinentes pour la prédiction courante en se basant sur un ensemble de *clés* représentant ces *valeurs*. Dans l'exemple de la traduction de langage les *valeurs* et *clés* sont les états cachés associés à chacun des mots de la phrase en entrée ($h_{enc,mot}$) et la *requête* pour traduire le prochain mot correspond à l'état caché du décodeur issu de la traduction du mot précédent.

Maintenant que la terminologie est généralisée, on se rend compte que l'on peut éventuellement utiliser les mécanismes d'attention avec d'autres types d'architectures que l'encodeur/décodeur *LSTM*. On a aussi la liberté d'associer les informations qui nous semblent pertinentes aux *valeurs*, *clés* et *requêtes*. Il est important de noter que la sémantique portée par le module d'attention dépendra ainsi des informations utilisées comme valeurs, clés et requêtes. La sémantique portée par un module d'attention pourra être généralisée comme suit : étant donné une *requête* on juge la pertinence d'un ensemble de *valeurs* en se basant sur un ensemble de *clés* appairées à ces *valeurs*. En revanche étant donné l'aspect boite noire des modèles en apprentissage profond, on n'a aucune garantie qu'une telle sémantique est

effectivement exprimée au travers d'une telle architecture. Il s'agit plus d'une manière mentale de concevoir les mécanismes d'attention. Voilà qui résume l'idée générale derrière un mécanisme d'attention.

Dans la suite, on décrit plus en détail certains mécanismes d'attentions présents dans la littérature et utiles pour notre étude. On les décrit d'abord dans le contexte de leurs études respectives, puis on regarde comment ils peuvent être généralisés en utilisant la terminologie présentée plus haut.

$$S = score(Q, K) \tag{2.5}$$

$$W = softmax(S) \tag{2.6}$$

$$Att(Q, K, V) = W^T \cdot V \tag{2.7}$$

On définit ici, selon la terminologie généralisée le mécanisme dit de *soft-attention* proposé séparément par Bahdanau et al. [22] et Xu et al. [23] . Dans ce mécanisme, on associe à chaque *valeur* v de la matrice V un poids. Le poids exprime la pertinence relative de la *valeur* par rapport aux autres *valeurs*. Pour cette raison, la somme des poids est égale à un. Le poids associé à une valeur est obtenu avec une fonction de score prenant en entrée la matrice Q des *requêtes* et la matrice K des *clés*. Les scores sont regroupés dans une matrice S (équation 2.5). Pour que les scores respectifs de chacune des valeurs somment à un, on applique la fonction *softmax* sur ces derniers (équation 2.6). Enfin, le vecteur résultant du module d'attention correspond à la somme pondérée des vecteurs d'attributs des *valeurs* contenus dans la matrice V par leurs poids respectifs, regroupés dans la matrice W), et obtenus dans la fonction de score (équation 2.7). Il est important de remarquer que différentes fonction de scores peuvent être utilisées. Le mécanisme de *soft-attention* est différentiable et peut ainsi être entraîné facilement via une rétro-propagation classique. Maintenant que le cadre général est défini, on définit deux exemples d'application du mécanisme de *soft-attention*.

Le premier exemple proposé par Bahdanau et al. [22] reprend le problème de la traduction de langage illustré dans la figure 2.6. Le mécanisme d'attention s'intègre dans une architecture encodeur/décodeur *LSTM*. Le but du modèle est de traduire une phrase en entrée (séquence de mots) en une phrase en sortie correspondant à sa traduction dans une autre langue. La phrase en entrée est encodée par un encodeur *LSTM*. A chaque mot i de la phrase en entrée, est associé un encodage $h_{enc,i}$. Ces encodages correspondent à la fois aux *clés* et aux *valeurs*. L'état caché du décodeur est ensuite initialisé par le dernier encodage issu de l'encodeur.

Pour prédire le prochain mot de la phrase traduite, on utilise comme contexte de prédiction l'état caché courant du décodeur $h_{dec,t-1}$, c'est-à-dire celui ayant permis de générer le mot *t-1* dans la phrase en sortie. Il correspond à la *requête*. Les formules générales du module d'attention sont les mêmes que celles utilisées pour la description générale du module de *soft-attention*. On revient plus en détail sur les formules appliquées à cet exemple. Dans celui-ci, la fonction de score entre une *requête* et une *clé*, correspond à appliquer une fonction non-linéaire $f(.)$ (soit un *MLP*) sur la concaténation (notée [a ;b] pour deux vecteurs a et b) de cette *requête* et de cette *clé*. La fonction non-linéaire produit une valeur réelle $s_{i,t}$ (équation 2.9) correspondant au score associé à cette paire *clé/requête*. Le vecteur d'attention c_t issu du module d'attention correspond à la somme pondérée des encodages $h_{enc,i}$ par leurs poids $\alpha_{i,t}$ (équation 2.10). D'un point de vue sémantique, le module d'attention évalue la pertinence de chaque mot de la phrase en entrée étant donné une représentation de celui-ci et le contexte de prédiction. Le contexte de prédiction, grâce au mécanisme de mémorisation du *LSTM* condense à la fois les informations venant des états cachés de l'encodeur de la phrase en entrée ainsi que des mots déjà prédits par le décodeur.

$$s_{i,t} = f([h_{enc,i}; h_{dec,t-1}]) \tag{2.8}$$

$$\alpha_{i,t} = \frac{\exp(s_{i,t})}{\sum_{i=1}^{n} \exp(s_{i,t})} \tag{2.9}$$

$$c_t = \sum_{i=1}^{n} \alpha_{i,t} h_{enc,i} \tag{2.10}$$

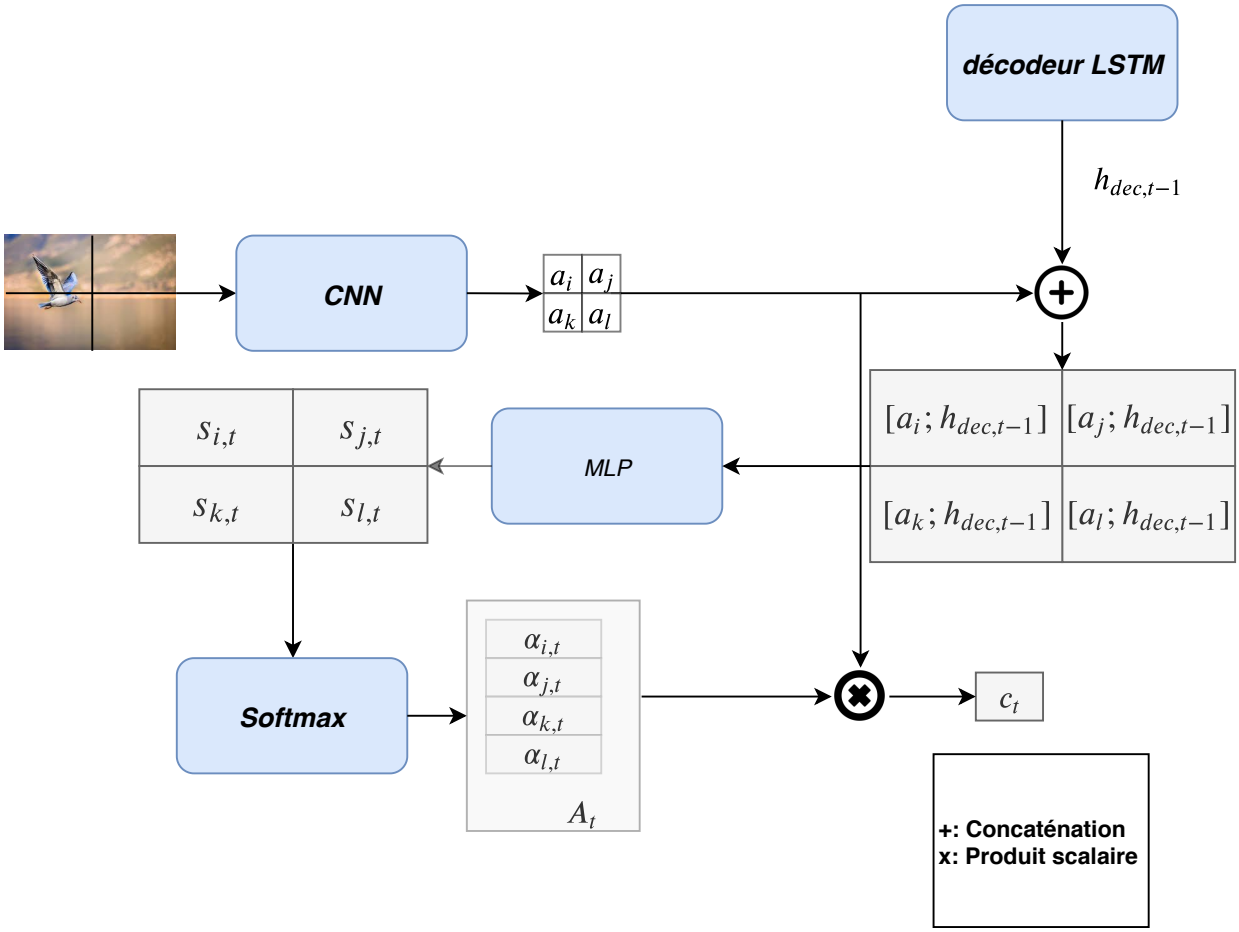


Figure 2.7 Mécanisme d'attention visuelle

Le deuxième exemple proposé par Xu et al. [23] a pour but d'associer à une image une phrase la décrivant. Le module d'attention, représenté dans la figure 2.7 est par conséquent appliqué à une image contrairement à l'exemple précédent où il était appliqué à une séquence. On peut appeler ce module d'attention *module d'attention visuelle*. L'image est passée dans un *CNN* pré-entrainé. Il nous permet d'obtenir un vecteur d'attributs a_i par partie de l'image. En sortie du *CNN* pré-entrainé, on obtient un grille de vecteurs d'attributs. La convolution est une opération qui conserve les dépendances spatiales. Ainsi, chaque cellule de la grille correspond à une partie de l'image. La dimension du vecteur d'attributs est donnée par le nombre de filtres de convolution utilisés dans la dernière couche de convolution du *CNN*. Ces vecteurs d'attributs sont utilisés à la fois comme *clés* et *valeurs*. Ensuite, comme dans l'exemple précédent, pour prédire le prochain mot de la description de l'image, on utilise l'état caché courant du décodeur $h_{dec,t-1}$ comme *requête*, c'est-à-dire comme contexte de prédiction. Dans cette étude, la fonction de score entre une *requête* et une *clé*, correspond à appliquer

une fonction non-linéaire $f(.)$ (soit un *MLP*) sur la concaténation de cette *requête* et de cette clé. Ici on concatène donc l'état caché courant du décodeur $h_{dec,t-1}$ et le vecteur d'attributs d'une partie de l'image a_i. La fonction non-linéaire produit une valeur réelle $s_{i,t}$ correspondant au score associé à cette paire clé/requête (équation 2.11). Le vecteur d'attention c_t issu du module d'attention correspond à la somme pondérée des vecteurs d'attributs a_i par leurs poids $\alpha_{i,t}$ (équation 2.13).

$$s_{i,t} = f([a_i; h_{dec,t-1}]) \tag{2.11}$$

$$\alpha_{i,t} = \frac{\exp(s_{i,t})}{\sum_{i=1}^{n} \exp(s_{i,t})} \tag{2.12}$$

$$c_t = \sum_{i=1}^{n} \alpha_{i,t} a_i \tag{2.13}$$

Il est courant que les clés et les valeurs renvoient aux mêmes objets. Quand les clés, les valeurs et les requêtes renvoient toutes aux mêmes objets, on parle d'*auto-attention* dans le sens où le contexte utilisé pour définir la pertinence d'un objet est donné par l'objet lui-même.

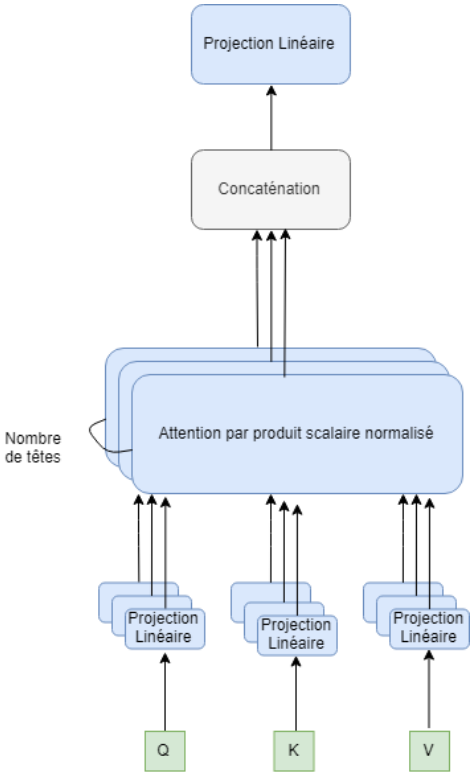


Figure 2.8 Mécanisme d'attention multi-tête adapté de [4]

On peut remarquer que ces mécanismes d'attention utilisent tout deux une architecture proche de l'encodeur/décodeur *LSTM* appartenant au paradigme récurrent. C'est le cas pour la grande majorité des mécanismes proposés dans la littérature, car ces modèles ont été développés pour être utilisés dans le traitement du langage naturel, donc avec des séquences. Vaswani et al. [4] proposent le premier (et unique) mécanisme d'attention n'utilisant pas cette architecture. Leur approche présente un modèle encodeur/décodeur pour la traduction de langage utilisant exclusivement des modules d'attention. L'architecture repose sur le concept d'attention multi-tête (figure 2.8) fonctionnant de la manière suivante : soit une phrase en entrée contenant les mots m_i. Pour chaque mot de la phrase, on applique un module d'auto-attention sur tous les mots de la phrase. Chaque mot est à la fois *valeur*, *clé* et *requête*. On applique sur chaque mot un module d'auto-attention. C'est dire que étant donné ce mot, on associe un poids à tous les mots de la phrase y compris lui-même. Cela permet de modéliser le sens de ce mot en fonction des interactions qu'il entretient avec les autres mots de la phrase. Les modules d'auto-attention sont appliqués en parallèle pour chaque mot. Pour rappel, les *clés*, *valeurs* et *requêtes* renvoient au même ensemble d'objets dans le cadre de l'auto-attention. De plus, pour accélérer le calcul, la fonction de score utilisée dans le module d'auto-attention est un produit scalaire normalisé entre une *clé* et une *requête*. L'opération du produit scalaire ne contient pas de poids à entraîner par rétro-propagation du gradient. Pour entraîner le réseau de neurone à donner un score à une paire *clé*/*requête*, on projette respectivement les *clés*, *requêtes* et *valeurs* dans trois espaces distincts de même dimension p en utilisant respectivement trois couches de réseau de neurone complètement connectée. Plutôt que de calculer de manière séquentielle tous les modules d'auto-attention, on regroupe les vecteurs d'attributs *clés*, *valeurs* et *requêtes* dans les matrices Q, K et V respectivement. Projeter les *requêtes* dans un espace revient à multiplier la matrice Q par le vecteur de poids W_q de la couche complètement connectée correspondante. Il en va de même pour les *clés* et les *valeurs*. Les scores pour chaque module d'auto-attention sont obtenus par un produit scalaire normalisé entre les matrices Q et V (équation 2.14). On obtient ainsi une matrice de score S de dimension $n \cdot n$ si la phrase contient n mots. Sur la première ligne, on trouve les scores associés par le premier mot à chacun des autres mots de la phrase. On applique la fonction *softmax* sur chacune des lignes de la matrice de score pour qu'une ligne somme à un. Pour faire la somme pondérée des *valeurs* pour chaque module d'attention, il ne reste plus qu'à faire le produit matriciel entre la matrice de poids et la matrice des valeurs V (équation 2.15).

L'ensemble des modules d'auto-attention pour une phrase constitue une tête d'attention. Pour capturer plusieurs variétés d'interactions entre les mots, ils proposent de calculer en parallèle plusieurs têtes d'auto-attention et de concaténer leurs résultats. Dans le cas de

l'auto-attention, Q, K et V prennent le même ensemble de valeurs à savoir tous les mots de la phrase. Cependant, calculer plusieurs modules d'attention pour chacun des mots de la phrase peut s'avérer coûteux en temps de calcul si on utilise une fonction de score inadaptée. Utiliser le produit scalaire normalisé comme fonction de score permet de paralléliser les calculs et ainsi de les accélérer. Les poids appris par le modèle permettent de différencier les informations apportées par les différentes têtes d'attention.

$$S = score(Q, K) = \frac{(W_q \cdot Q^T) \cdot (W_k \cdot K^T)^T}{\sqrt{p}} \tag{2.14}$$

$$att(Q, K, V) = softmax(S) \cdot V \tag{2.15}$$

avec $\cdot$ le produit matriciel.

Utiliser plusieurs têtes d'auto-attention permet de focaliser l'attention sur différents aspects de relations entre valeurs. Cela est permis par la fonction de score très rapide à calculer et facile à paralléliser grâce au produit scalaire.

2.1.8 Réseaux génératifs adversariaux

Les réseaux génératifs adversariaux (GAN), ont été proposés par Goodfellow et al. [24]. Un GAN (figure 2.9) est composé de deux réseaux de neurones qui s'affrontent. Le générateur apprend à imiter une distribution de données et à en générer des échantillons aléatoirement. Le discriminateur apprend à distinguer les échantillons provenant de la distribution originale des imitations générées par le générateur. Le but du générateur est de tromper le discriminateur en générant des échantillons suffisamment réalistes pour que ce dernier ne puisse pas les distinguer de ceux de la distribution d'origine. Les réseaux sont entraînés en alternance au sein d'une même itération. On commence par entraîner le discriminateur, puis on fixe ses poids et on entraîne le générateur contre ce nouveau discriminateur et ainsi de suite. Le générateur apprend à transformer une distribution quelconque (loi normale par exemple) à la distribution de l'ensemble de données, permettant de générer un échantillon de la distribution des données à partir d'un échantillon d'une distribution connue. Le discriminateur permet d'assurer le réalisme de cette association de distribution. La fonction de perte permettant l'apprentissage du générateur et du discriminateur se base sur le résultat de la classification du discriminateur.

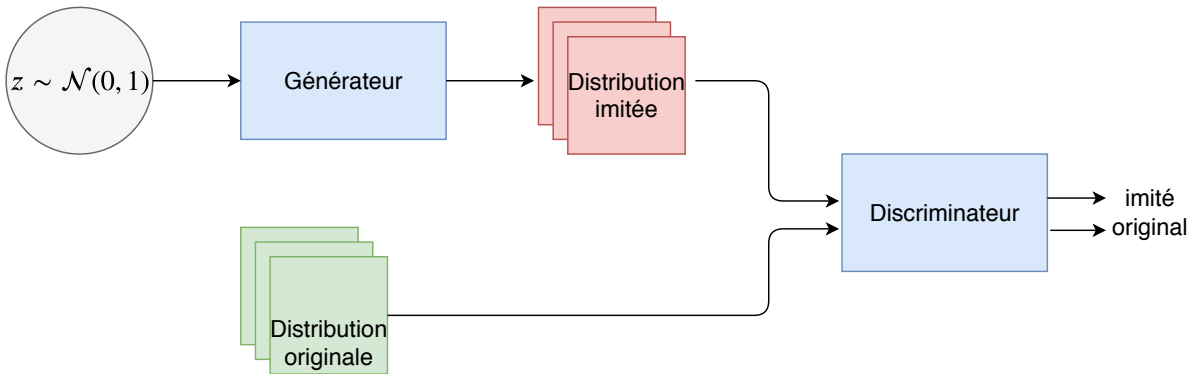


Figure 2.9 Réseau génératif adversariel

2.2 Travaux relatifs à la tâche de prédiction de trajectoire

Dans cette deuxième partie de la revue de littérature, on propose une vue d'ensemble des différentes approches proposées dans la littérature pour mener à bien la tâche de prédiction de trajectoire.

2.2.1 Les méthodes antérieures à l'apprentissage profond

Dans leur taxonomie de 2014, Lefèvre et al. [25] recensent la majorité des travaux effectués jusqu'à lors et les regroupent en trois catégories de modèles : les modèles basés sur la dynamique (ou *Physics-based*), les modèles basés sur les patrons de mouvement (ou *Maneuver-based*) et les modèles prenant en compte les interactions (ou *Interaction-aware*).

Les modèles dits *Physics-based* effectuent une prédiction en considérant que le mouvement de l'agent est uniquement régi par les lois physiques de la dynamique (les forces exercées sur le véhicule) ou les lois de la cinétique (positions, vitesses et accélérations). Les modèles cinétiques les plus simples sont ceux à vitesse ou accélération constante, qui considèrent que le véhicule va conserver ses caractéristiques cinétiques au cours de la période de prédiction. Plus généralement, dans ce paradigme, on considère l'évolution d'un véhicule comme une série de transitions d'un état à l'autre. On connait les caractéristiques de l'état courant et le modèle a pour rôle d'exprimer la transition d'un état à l'autre. On peut citer le filtre de Kalman utilisé par Ammoun et al. [26], Welch et al. [27] et Lin et al. [28] qui permet d'effectuer une prédiction assortie d'une incertitude gaussienne, sous l'hypothèse de la nature gaussienne de cette incertitude et de la linéarité du phénomène de transition. Si ces deux hypothèses ne tiennent pas, la méthode de Monte-Carlo est utilisée par Tran et al. [7] et Eidehall et al. [29], pour

estimer la distribution de la trajectoire future. La distribution est approximée à partir d'un échantillonnage aléatoire pouvant être soumis à des contraintes concernant l'environnement de prédiction. Lefèvre et al. [25] énoncent les limites de cette catégorie de modèles. La prise en compte du contexte de la prédiction est limité. Ne sont pris en compte ni l'environnement social ou spatial, ni la décision éventuelle de l'agent de changer de manœuvre, ni la volonté de l'agent (destination finale). Ces modèles sont donc limités à la prédiction à court-terme.

Les modèles dits *Maneuver-based* considèrent l'agent comme effectuant une suite de manœuvres (aller tout-droit, tourner à gauche...) sans considérer le contexte global de cette suite de décisions. On cherche donc à identifier la manœuvre entreprise par l'agent, cette dernière nous permettant d'effectuer une prédiction quant à sa future trajectoire. La principale approche de cette catégorie est l'approche des prototypes ou patrons de mouvements. Étant donné une scène, on dispose de l'historique des trajectoires des agents l'ayant traversée au cours d'un certain laps de temps. Ces trajectoires sont séparées en groupes, selon une mesure de similarité des trajectoires correspondant à un type de manœuvre (Li et al. [30]). Chaque groupe correspond à un type de manœuvre si les mesures de similarité et les paramètres de regroupement sont bien choisis. Deux éléments primordiaux dans cette méthode sont, la mesure de distance entre deux trajectoires ainsi que l'algorithme de regroupement choisi. Les distances les plus utilisées sont *Longest Common Subsequence (LCSS)* et *Dynamic Time Warping (DTW)* qui permettent de mesurer la distance entre deux séries temporelles de longueurs différentes tout en tolérant un peu de bruit dans les données. Ces métriques et divers algorithmes de regroupements sont comparés par Morris et al. [31] qui concluent que les algorithmes de regroupement sont équivalents selon la métrique *Correct Clustering Rate* proposée par Zhang et al. [32]. Pour mieux tenir compte des variations de dynamique des trajectoires dans un même groupe, plusieurs regroupements consécutifs peuvent être effectués, comme dans Hu et al. [33] où un regroupement basé sur les vitesses suit un regroupement basé sur les positions. Une fois les prototypes obtenus, ils sont représentés de diverses manières pour prendre en compte les variations de dynamique mentionnées plus haut. Un groupe peut-être représenté par son centroïde, ou par la moyenne et l'écart-type des trajectoires qui le composent (Large et al. [34]), ou par un sous-ensemble de ces trajectoires ou enfin par un modèle probabiliste entrainé sur les trajectoires qui le composent.

Les processus gaussiens, utilisés par Tran et al. [7] et Kim et al. [35] considèrent que les trajectoires d'un groupe sont générées par une distribution gaussienne. On estime les paramètres de cette distribution par régression de processus gaussiens. La représentation probabiliste permet de prendre en compte partiellement la diversité de déplacements et de dynamiques au sein d'un même groupe. L'entraînement de ces modèles est coûteux (Lefèvre et al. [25]), et ils ne peuvent être entraînés que sur des ensembles de données de petite taille

(plusieurs centaines de trajectoires). Une fois les prototypes et leurs représentations obtenues, on prédit la future trajectoire en associant la trajectoire partielle de l'agent considéré à son prototype le plus proche en terme de distance Morris et al. [31] ou le plus probable pour les modèles probabilistes (Joseph et al. [36]). Morris et al. [8] proposent un framework complet reposant sur le concept de prototypes pour une scène donnée.

Une alternative aux prototypes est de définir manuellement, pour une scène, les différentes manœuvres possibles puis d'entraîner un classificateur sur ces dernières (e.g. *Support Vector Machine (SVM)*, *Multi-Layer Perceptron (MLP)*) comme Aoude et al. [37] à partir d'attributs extraits manuellement. Chaque manœuvre peut être représentée par un processus gaussien comme précédemment. Ces modèles sont plus intéressants que les *Physics-based* car ils considèrent d'une certaine manière l'intention de l'agent. Ils restent néanmoins limités. Les prototypes sont extraits pour une unique scène ce qui rend le modèle non-transférable sur une autre. De plus, les interactions entre agents ne sont pas prises en compte dans le choix de la manœuvre effectuée. Enfin, les modèles basés sur la classification d'intention nécessitent l'étiquetage manuel des différents types de mouvements ainsi que le choix des attributs jugées pertinentes pour la tâche de classification (Lefèvre et al. [25]).

Les modèles *Interaction-aware* étaient peu nombreux en 2014. Des modèles paramétriques comme *social forces* de Helbing et al. [38], définissent manuellement des forces régissant les interactions d'un agent avec son environnement. Les forces sont définies par des paramètres. Par exemple, la force de répulsion entre deux agents ou la force d'attraction d'un agent vers son objectif dans la scène. Si ces modèles sont plutôt performants, leurs principaux défauts sont la définition manuelle de toutes les forces et de leurs paramètres associés ainsi que la difficulté à les calibrer.

Des approches basées sur les prototypes tentent de prendre en compte les interactions entre agents. Yoo et al. [39] proposent un deuxième regroupement des prototypes pour apprendre lesquels ont tendance à se produire en même temps, pouvant ainsi prédire des manœuvres d'évitement entre deux agents. Lawitzky et al. [40] regroupent les paires de prototypes se produisant fréquemment ensemble. Au moment d'associer une paire de prototypes à une paire de sous-trajectoires, on pénalise les possibilités n'étant pas comprises dans les paires se produisant fréquemment.

Par ailleurs, les réseaux bayésiens dynamiques (*DBN*) et leur cas particulier Chaine de Markov Cachées (*HMM*), sont les plus utilisés. Les réseaux bayésiens sont des modèles probabilistes représentant plusieurs variables aléatoires ainsi que leurs dépendances sous forme de graphe orienté. A partir de ce graphe, on peut calculer les probabilités conditionnelles entre les variables aléatoires. On parle de *DBN*, quand la dépendance des variables aléatoires est

étendue le long de la dimension temporelle. (Une variable aléatoire à l'instant t est conditionnée sur elle-même et d'autres aux instants précédents). Les *HMM* sont un cas particulier des *DBN* respectant la propriété de Markov, c'est-à-dire qu'une variable aléatoire à l'instant t est conditionnée uniquement sur sa valeur à l'instant *t-1* et non sur les instants précédents. Dans le cadre de la prédiction, ces modèles permettent d'exprimer les dépendances entre les véhicules et d'introduire des contraintes liées à l'environnement par l'intermédiaire de variables aléatoires. Les *HMM* sont utilisés par Firl et al. [41] et Meyer et al. [42] pour modéliser différentes manœuvres complexes comme un dépassement sur l'autoroute. Les auteurs entraînent un modèle par type de manœuvre. Gindele et al. [43] proposent un modèle bayésien prenant en compte les contraintes de l'environnement, ce modèle est appris automatiquement.

Les modèles *Interaction-aware* sont les plus intéressants, puisqu'ils permettent de faire des prédictions en prenant en compte les interactions entre véhicules et certaines caractéristiques de l'environnement permettant ainsi des prédictions plus fiables. Cependant, pour prendre en compte toutes les paires d'agents deux à deux, ces modèles sont lourds en temps de calcul. Au regard de Lefèvre et al. [25], la recherche commence à développer de nouvelles approches *interaction-aware* dont l'objectif est de considérer toutes les interactions agent/agent et agent/espace au sein d'une scène, tout en conservant un temps de calcul raisonnable. Enfin, une partie du domaine s'intéresse à entraîner des modèles pouvant généraliser à une nouvelle scène dont ils n'ont jamais observé l'historique des trajectoires. A partir de 2014, la majorité des travaux se concentrent sur ces axes. Vermula et al. [44] construisent un voisinage local de l'agent dont on veut prédire la trajectoire. Il s'agit d'une grille dont chaque case recense le nombre d'agents qu'elle contient. Un processus gaussien est entrainé à partir de cette information et de la destination connue de l'agent. Le modèle est entraîné et testé sur une seule scène. Robicquet et al. [45] proposent une méthode similaire à celle des forces sociales pour les piétons de Helbing et al. [38]. Ils définissent une valeur de sensibilité sociale pour caractériser les différents profils de navigation possibles. Chaque profil de navigation est obtenu par regroupement et dispose d'une valeur de sensibilité sociale associée. Le modèle est entraîné et testé avec différents types d'agents sur plusieurs scènes. Balan et al. [9] et Coscia et al. [46] divisent une scène en cellules. Pour chaque cellule et chaque type d'agent, sont calculés un histogramme de direction, de vitesse et un score de popularité. On obtient ainsi une représentation de la manière dont la scène est traversée. Un *DBN* est ensuite entraîné à partir de ces informations. La trajectoire est prédite jusqu'à la sortie de l'agent de la scène. Un transfert de connaissance d'une scène à l'autre peut-être effectué par association de cellules. Le transfert conduit à 70% d'erreur supplémentaire par rapport à l'entraînement sur la scène que l'on veut tester. Ce type de méthode est performant pour de la prédiction à long-terme en entraînant sur unique scène.

L'ensemble de ces travaux propose des idées intéressantes mais ils sont tous incomplets, certains ne prennent en compte que les interactions agent/agent (Vemula et al. [44] et [45]), d'autres les interactions humain/espace (Vemula et al. [44] et Coscia et al. [46]), un seul type d'agent (Vemula et al. [44], Ballan et al. [9] et Coscia et al. [46]). Leur principale faiblesse est leur spécificité à une scène et la difficulté à transférer la connaissance.

2.2.2 L'apprentissage profond

En parallèle, fort de son succès dans différents domaines comme la vision par ordinateur avec les *CNN* de Lecun et al. [10] ou le traitement du langage naturel avec les *RNN*, l'apprentissage profond est employé pour tenter de résoudre les problèmes persistants des diverses approches de prédiction de trajectoire.

Les avantages principaux de l'apprentissage profond sont les suivants :

— Pouvoir généraliser la prédiction à une scène jamais vue.
— Ne pas définir manuellement les caractéristiques à utiliser.

Plusieurs approches ont vu le jour qui peuvent être réparties en différentes catégories :

1. Une partie se concentre à prendre en compte les interactions agent/agent. Alahi et al. [47], Bartoli et al. [48], Lee et al. [11], Varshneya et al. [49], Deo et al. [50] et Gupta et al. [51] définissent un voisinage local sous forme de grille centré autour de l'agent principal, et l'utilisent comme caractéristiques supplémentaires en entrée d'un *RNN* encodeur/décodeur. Radwan et al. [52] utilisent un *CNN* prenant comme entrée un vecteur à trois dimensions (nombre d'agents, durée d'observation, nombre de coordonnées spatiales) et effectuent la prédiction simultanée des futures trajectoires de tous les agents.
2. Une autre partie étudie les interactions agent/espace. Bartoli et al. [48] définissent des objets d'intérêt dans la scène et utilisent une grille similaire à Alahi et al. [47], Bartoli et al. [48], Lee et al. [11], Varshneya et al. [49], Deo et al. [50] et Gupta et al. [51] pour prendre en compte les interactions de l'agent avec les objets de son voisinage. Manh et al. [53] divisent la scène en cellules et associe un *RNN* par cellule permettant de retenir la dynamique des trajectoires traversant cette partie de la scène et d'orienter la future trajectoire de l'agent étant donnée la mémoire de cette cellule. Lee et al. [11] apprennent les caractéristiques spatiales de la scène en extrayant les attributs de l'image de la scène vue de dessus avec un *CNN*. Varshneya et al. [49] divisent l'image en cellules et prédisent pour chaque cellule la probabilité que l'agent s'y rende en utilisant des *CNN*.

3. L'attention est utilisée pour les interactions agent/agent, pour déterminer quels agents sont les plus pertinents à prendre en compte. Fernando et al. [54] associent un poids à chaque agent voisin et à chaque instant de la période d'observation. Ils associent manuellement un poids inversement proportionnel à la distance entre l'agent et le voisin i à l'instant t. Sadeghian et al. [6] et Vemula et al. [55] laissent le réseau se charger automatiquement d'associer un poids à chacune des représentations des trajectoires passées des agents voisins. L'attention est aussi utilisée dans les interactions agent/espace. Sadeghian et al. [6] extraient un vecteur de caractéristiques par partie de l'image de la scène vue de dessus. Ils associent ensuite automatiquement un poids à chaque vecteur. Sadeghian et al. [56] utilisent le même module d'attention que [6] et le combinent avec un autre module. Le réseau définit automatiquement les coordonnées et les paramètres d'une grille de filtres de convolution à appliquer sur l'image. Ce procédé permet de se concentrer sur une unique partie de l'image à l'inverse des procédés précédents prenant en compte chaque partie de l'image proportionnellement à son poids.
4. Des approches plus visuelles comme Huang et al. [57], Baumann et al. [58], Hoermann et al. [59] et Jeon et al. [60] représentent la scène sous forme d'une grille comportant diverses informations et effectuent les prédictions à partir d'un *CNN*. Les trajectoires sont représentées de manière visuelle en indiquant par quelle cellule l'agent est passé.
5. Il existe des modèles génératifs proposés par Lee et al. [11], Gupta et al. [51], Sadeghian et al. [6], Hug et al. [12]. Pour un même contexte, le modèle effectuera différentes prédictions. Le modèle est entraîné pour associer la distribution des données à une distribution connue (loi normale centrée réduite par exemple). Une fois le modèle entraîné, il est possible de générer un échantillon de la loi connue et de le transformer, grâce au modèle, en échantillon de la distribution des données. Hug et al. [12] utilisent le modèle de Graves et al. [61] pour la génération d'écriture manuscrite. Lee et al. [11] utilisent le *Conditional Variational Auto Encoder (CVAE)* de Kingma et al. [62]. Gupta et al. [51] et Sadeghian et al. [6] utilisent le *GAN* de [24]. Les modèles génératifs permettent de prendre en compte le fait que le problème de prédiction de trajectoire ne soit pas un problème déterministe, c'est-à-dire que pour un contexte de prédiction donné, plusieurs prédictions correctes sont possibles.
6. Si les modèles complexes montrent un intérêt pour les résultats de prédiction, des modèles plus naïfs ne prenant pas en compte les interactions agent/agent et agent/espace donnent de bons résultats selon les métriques de ces articles. Nikhil et al. [5] utilisent un *CNN* sur la dimension spatiale pour prédire la future trajectoire. Becker et al. [63] étudient différents modèles naïfs et trouvent que tous ont des performances similaires,

le meilleur étant le *LSTM-MLP*. Ils encodent la trajectoire passée d'un agent avec un *LSTM*, puis prédisent simultanément toutes les futures coordonnées avec un *MLP*. Un tel modèle impose de prédire des trajectoires de tailles fixes.

Les différentes approches en apprentissage profond semblent prometteuses et permettent notamment de prendre en compte le contexte social et le contexte spatial. Les mécanismes d'attention permettent de prendre en compte ces mêmes informations tout en évitant de devoir définir manuellement les attributs à considérer à l'inverse du tenseur de réduction social de Alahi et al. [47]. Cependant, les modalités d'évaluation des différentes approches rend difficile leur interprétation et leur comparaison, de telle sorte qu'il est difficile de vraiment apprécier l'impact des contributions proposées par chaque étude. Nous reviendrons plus en détail sur le problème de l'évaluations des performances des différentes approches dans la section suivante.

D'autre part, les approches utilisant les mécanismes d'attention sont peu nombreuses et utilisent toutes le même procédé transposant naïvement les architectures issues du traitement du langage naturel sans s'interroger sur la sémantique portée par ces mécanismes d'attention et la pertinence de les utiliser tel quel. Nous étudierons de manière plus approfondie ces mécanismes d'attention dans la section suivante.

2.3 Les approches naïves pour la prédiction de trajectoire

Dans cette partie et la suivante, on revient plus en détail sur les approches naïves rencontrées dans la littérature. Pour rappel, les approches dites naïves ne prennent en compte que la trajectoire observée de l'agent principal pour faire une prédiction. On y mentionne sans les détailler les ensembles de données utilisés par les études. Ils sont décrits plus en détail dans la section 2.5.

Hug et al. [12] proposent *LSTM-MDL*. Ils s'inspirent de l'architecture de Graves et al. [61] en la modifiant légèrement. Ils prennent en entrée les positions absolues plutôt que relatives et prédisent en sortie les paramètres d'un mélange de gaussiennes exprimant la distribution des positions relatives pour la prochaine position. Le nombre d'unités de temps observées n'est pas clair. En revanche, la prédiction est effectuée pour le nombre d'unités de temps restantes dans la trajectoire réelle jusqu'à la sortie de la scène par l'agent. Ils étudient la question suivante : pourquoi un modèle génératif pouvant représenter toute la variabilité spatio-temporelle d'une scène obtient de moins bonnes performances qu'un modèle de prédiction simple à vitesse constante sur les scènes plus complexes ? Intuitivement on s'attendrait à l'inverse. Ils l'expliquent de la manière suivante :

— Plus la scène est complexe, plus le nombre de possibilités spatio-temporelles (ou manœuvres) augmente.
— Si le modèle est capable de représenter cette variabilité, alors il a plus de chances de se tromper.

Par exemple si l'agent principal doit traverser un embranchement à trois branches, le modèle complexe en faisant un choix parmi ces trois possibilités a 66 % de chance de se tromper complètement. L'effet mentionné est a priori d'autant plus fort que la prédiction est effectuée à long-terme.

Nikhil et al. [5] proposent un *CNN* pour la tâche de prédiction. Il s'agit d'un prédicteur naïf ne prenant en considération que la trajectoire passée de l'agent principal. Le modèle est représenté dans la figure 2.10. Les positions de la trajectoire observée (x_i, y_i) sont d'abord projetées dans un espace de dimension supérieure, il s'agit simplement de passer ces coordonnées dans un *MLP*. On obtient un vecteur p_i par position. Ensuite, un simple *CNN* est appliqué le long de la dimension temporelle. Pour que sa longueur soit la même avant et après la convolution, on ajoute des zéros de part et d'autre de la trajectoire de manière symétrique. Les différents vecteurs résultant de la convolution f_i sont ensuite concaténés et passés dans un *MLP*, comme dans l'approche de Becker et al. [63] pour prédire simultanément les futures positions de l'agent. Dans un *CNN*, contrairement au *LSTM*, les opérations sont effectuées en parallèle et non de manière séquentielle. Ce modèle propose donc de réduire le temps de prédiction.
Les performances de ce modèle sont évaluées sur les ensembles de données *UCY* [64] et *ETH* [65] selon le principe de validation croisée par scène. La période d'observation est de huit unités de temps, contre douze pour la période de prédiction. Le modèle est comparé à d'autres prenant en compte ou non des interactions agent/agent et agent/espace et obtient de meilleurs résultats sur cet ensemble de données. Le temps de calcul pour une trajectoire est comparé à l'approche de Gupta et al. [51] et à un *LSTM* classique. Le *CNN* conduit à une division par trente et par cinq, respectivement, du temps de prédiction.

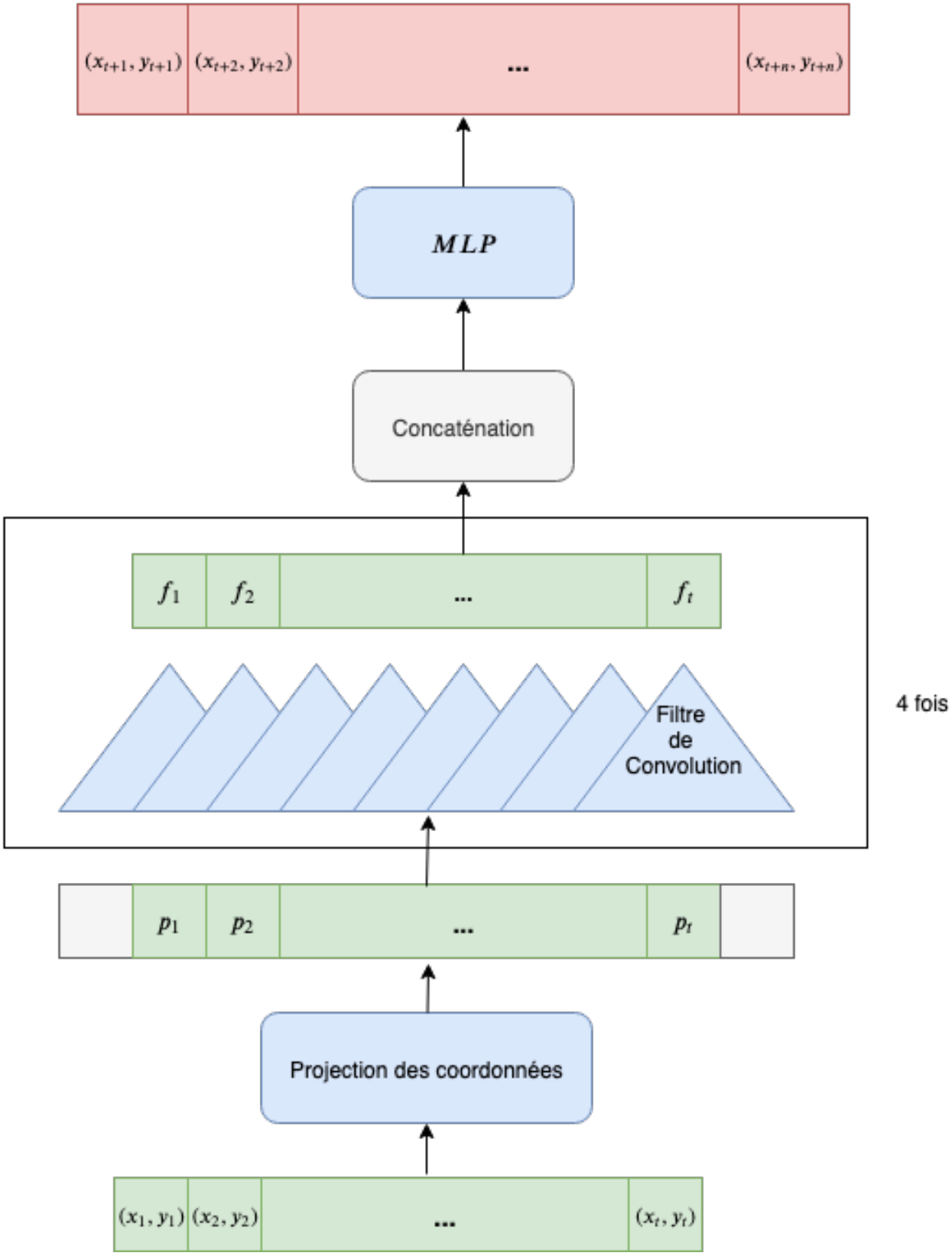


Figure 2.10 CNN pour la prédiction de trajectoire adapté de [5]

Becker et al. [63] proposent le modèle encodeur-récurrent suivi d'un *MLP*(ou *Recurrent-Encoder with a Dense layer stacked on top (RED)*. Dans ce travail, ils proposent d'essayer d'obtenir les meilleures performances possibles à partir d'architectures naïves, c'est-à-dire qui ne considèrent pas des informations autres que la trajectoire passée de l'agent. Ils testent différentes architectures comme le *LSTM* encodeur/décodeur, le *TCN*, ou le *LSTM-MLP*. Ils concluent que ces différentes approches sont globalement équivalentes. Différentes modalités de prédiction sont étudiées. Ils proposent de prendre en entrée les positions relatives plutôt que les positions absolues. Et de prédire les positions relatives par rapport au dernier point de la trajectoire observée plutôt que les futures positions absolues. Ils le justifient par le fait que la distribution des positions relatives est bien plus facile à apprendre que celle des positions absolues et aussi plus facilement transférables d'une scène à l'autre. Enfin, prédire toutes les futures positions en une fois serait préférable à les prédire de manière récursive afin de minimiser l'erreur cumulée sur l'ensemble des prédictions. Le modèle est testé sur le *Stanford Drone Dataset (SDD)* en conservant un sous-ensemble de scènes pour le test. Il existe plusieurs variations pour l'ensemble de données *SDD*, une contenant divers types d'agents et une conservant uniquement les piétons. Ici, et dans toutes les approches mentionnées dans la revue de littérature, les méthodes sont testées uniquement sur les trajectoires des piétons. Pour les différents critères proposés relatifs à l'utilisation des positions relatives, aucune étude ablative ne permet de valider indépendamment leur intérêt. Le modèle obtient de meilleures performances que des modèles plus complexes.

Nous reviendrons dans la section 2.5 sur le fait que les approches naïves obtiennent parfois de meilleurs résultats que les approches complexes.

2.4 Les mécanismes d'attention pour la prédiction de trajectoires

Peu d'études concernant les mécanismes d'attention pour la prédiction de trajectoires sont disponibles dans la littérature. Dans cette partie nous les décrivons plus en détail. Fernando et al. [54] proposent *Soft+Hardwired attention*. Le modèle repose sur une architecture *LSTM* encodeur/décodeur. Dans un premier temps, la trajectoire observée de l'agent principal et celles de ses voisins sont passées dans l'encodeur *LSTM*. On dispose donc d'un état caché (encodage) par unité de temps, par agent dans la scène. Pour prédire la position de l'agent principal lors de la prochaine unité de temps, on utilise deux modules d'attention. Le premier est un module de *soft-attention* appliqué sur les états cachés de l'agent principal. À partir de l'état caché du décodeur *LSTM* pour l'unité de temps courante, un poids est associé à chaque état caché. Le résultat du module d'attention est la somme, pondérée par ces poids, des états cachés. Le deuxième module d'attention est similaire à un module de *soft-attention*

à la différence que les poids sont définis manuellement par une règle. Pour l'état caché d'un agent voisin lors d'une unité de temps de la période d'observation, on associe un poids inversement proportionnel à la distance entre l'agent principal et le voisin pour cette unité de temps. Cette règle a pour but d'indexer l'importance d'un voisin sur sa proximité à l'agent principal et de réduire le temps de calcul nécessaire aux poids d'attention. Les résultats des deux modules d'attention sont ensuite concaténés et utilisés comme entrée dans le décodeur afin de prédire la future position de l'agent principal. L'opération est répétée pour chaque prédiction.

Dans un deuxième temps, les trajectoires de l'ensemble d'entraînement sont regroupées par points d'entrée et de sortie de la scène, dans le but de prendre en compte les différentes modalités de déplacement de la scène. Un modèle est entraîné pour chacune de ces modalités. Considérer tous les états cachés de chaque agent, permettrait de mieux saisir l'intention générale de l'agent, et de moins être dans la réaction comme dans *social-LSTM* où l'on considère uniquement le dernier état caché. Le modèle est testé sur les ensembles de données *New-York Grand Central* de Zhou et al. [66] et *Edinburgh Informatics Forum* de Fisher [67]. Les trajectoires sont observées pendant vingt unités de temps, et prédites pour la même durée. Une étude ablative est réalisée, mettant en avant l'intérêt de l'attention pour les interactions agent/agent ainsi que l'intérêt d'entraîner un modèle par modalité de déplacement. L'intérêt de considérer tous les encodages de la trajectoire d'un agent n'est pas démontré. De plus le regroupement des trajectoires par modalité de déplacement empêche le modèle d'être utilisé sur une nouvelle scène sans informations additionnelles.

Sadeghian et al. [56] proposent *Car-Net*. Le modèle utilise un simple *LSTM*. Il utilise deux modules d'attention spatiale pour considérer les interactions agent/espace. Dans un premier temps, un *CNN* pré-entrainé et raffiné pour la tâche de segmentation d'image est utilisé pour extraire un vecteur d'attributs par partie de l'image. Un module d'attention visuelle est appliqué sur cet ensemble de vecteurs résultant en un vecteur d' attention spatiale. Ce module permet de prendre en compte chaque partie de l'image proportionnellement à son importance pour la prédiction de la prochaine position de l'agent principal. Un module d'attention à source unique de type grille de gaussienne (Gregor et al. [68]) est appliqué à l'ensemble de l'image. Ce module permet de se concentrer particulièrement sur une seule partie de l'image, tout en contrôlant le zoom appliqué à cette partie de l'image. Les résultats des deux modules d'attention et les coordonnées de la dernière position de l'agent principal sont concaténés et fournis en entrée au décodeur. L'opération est répétée pour chacune des futures positions. En d'autres termes, pour prédire la position à l'instant *t+1*, on utilise deux modules d'attention. Les poids fournis par chacun de ces modules sont obtenus comme l'application d'une fonction sur l'ensemble des vecteurs d'attributs de l'image de la scène et

l'état caché du décodeur à l'instant t. Les poids d'attentions sont combinés avec les vecteurs d'attributs de l'image de la scène pour chacun des modules d'attention.

Le modèle est testé sur *SDD*, en conservant un sous-ensemble de scènes pour le test. La période d'observation est de huit unités de temps, la période de prédiction de douze. Une étude ablative est réalisée mettant en avant l'intérêt de chaque composant pour la performance d'un point de vue quantitatif. Les performances sont reportées comme la moyenne en pixel des métriques sur l'ensemble des scènes de *SDD*. Etant donné que la hauteur du drone sur chaque scène n'est pas la même, reporter les résultats en pixel ne semble pas particulièrement pertinent pour représenter la qualité de prédiction du modèle.

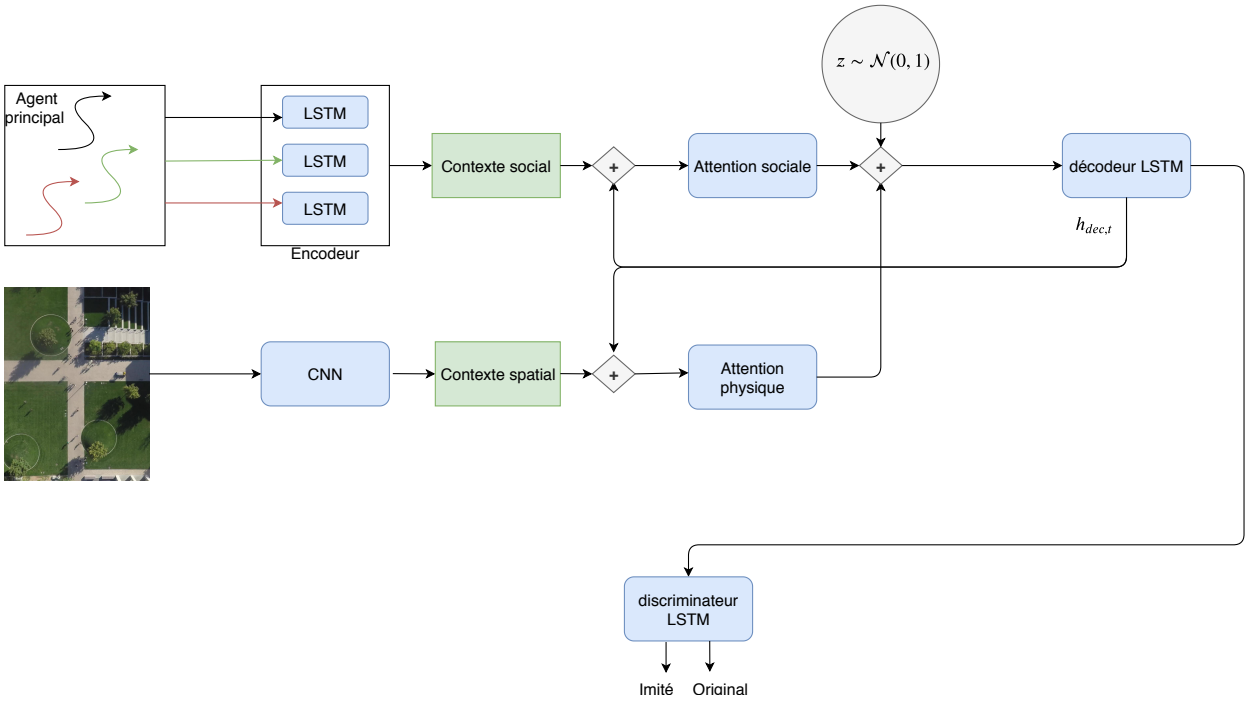


Figure 2.11 Architecture de sophie-GAN adapté de [6]

Sadeghian et al. [6] proposent *sophie-GAN*. L'architecture se base sur celle d'un *GAN LSTM* encodeur/décodeur. Le fonctionnement de ce modèle pour une prédiction à un instant donné est représenté sur la figure 2.11. Un module de *soft-attention* est utilisé pour les interactions agent/agent et un pour les interactions agent/espace respectivement.
Pour les interactions agent/espace, l'image de la scène en vue de dessus est passée dans un *CNN* pré-entrainé pour la segmentation d'image. Un vecteur d'attributs est obtenu par zone de l'image. Un module d'attention visuelle est appliqué, à partir de l'état caché du décodeur, sur ces vecteurs pour obtenir le vecteur d'attention spatiale. Pour les interactions agent/agent,

les trajectoires observées de l'agent principal et de ses voisins sont respectivement encodées par un encodeur *LSTM*. Pour l'agent principal, son vecteur de contexte social est le résultat d'un module de *soft-attention* sur les encodages de ses voisins, à partir de l'état caché du décodeur. Les vecteurs de contexte social et spatial sont concaténés à l'échantillon d'une variable aléatoire. Dans l'article, il n'est pas clair si le module d'attention est calculé une seule fois pour prédire l'ensemble de la trajectoire ou si il est recalculé pour la prédiction de chaque position. Dans le premier cas, étant donné que l'on utilise l'état caché du décodeur comme requête dans le module d'attention, et qu'il n'est pas précisé comment initialiser cet état caché, on ne sait pas si il porte le contexte de prédiction. Ca ne fait aucun sens si il est initialisé aléatoirement. Le deuxième cas est similaire à l'approche *Car-Net* des mêmes auteurs. On fait l'hypothèse que le module d'attention est recalculé à chaque unité de temps de prédiction. Ici on utilise conjointement la fonction de perte traditionnelle des *GAN* de Goodfellow et al. [24] pour entraîner le générateur à produire des trajectoires réalistes et la norme *L2* entre la trajectoire prédite et la trajectoire réelle pour entraîner le modèle à prédire une trajectoire proche en terme de positions de la trajectoire réelle. Le nombre maximal de voisins à considérer est fixé à trente-deux.

Les performances de ce modèle sont évaluées séparément sur l'ensemble de données *ETH/UCY* et *SDD*. Pour *ETH/UCY*, le principe de validation croisée avec omission de scène est utilisé. Pour *SDD*, le découpage de scènes proposé par le challenge *Trajnet* (Sadeghian et al. [13]) est utilisé. Le *SDD* utilisé ne contient que les trajectoires des piétons. Le modèle est entraîné sur une partie des scènes et testé sur le reste. Une étude ablative est réalisée permettant de mettre en avant l'intérêt de chaque module séparément et du modèle dans sa globalité. Il est intéressant de noter que le modèle n'apporte qu'une amélioration infime sur les scènes de l'ensemble de données *ETH/UCY*, et une bien plus importante sur les scènes de *SDD*. Une comparaison entre scènes simples et complexes de *SDD* est réalisée permettant de confirmer l'observation précédente. Une métrique d'évaluation de conflits est proposée afin d'évaluer l'impact de la prise en compte des interactions agent/agent sur la prédiction de collisions entre agents. On compte le nombre moyen de fois par unité de temps où deux agents sont séparés par une distance inférieure à dix centimètres. L'utilisation du module d'attention sociale permet de réduire le nombre moyen de collisions prédites par période de prédiction. Plusieurs réserves peuvent-être exprimées par rapport à cet article :

— Comme dans Car-Net, les résultats pour *SDD* sont reportés comme la moyenne en pixel des métriques d'évaluation sur l'ensemble des scènes, ce qui est problématique puisque la métrique mélange des erreurs de différentes tailles.

— Le modèle ignore les agents autre que les piétons et limite le nombre d'agents pris en compte à trente-deux. *SDD* contient presque autant de cyclistes que de piétons et

un nombre maximal d'agents par unité de temps supérieur à cent. Conserver uniquement les trajectoires des piétons nuit à la richesse et à la complexité des interactions proposées.

— Le discriminateur du *GAN* ne prends pas en compte le contexte social et spatial pour déterminer la véracité d'une trajectoire alors que dans un *GAN* conditionnel [69], il faut conditionner à la fois sur l'entrée du générateur et du discriminateur.

— Le *CNN* pré-entrainé est le *VGG-net* de Simonyan et al. [19]. Il est très volumineux, soit plusieurs millions de paramètres contre seulement quelques dizaines de milliers pour le reste de l'architecture.

La littérature propose donc trois approches s'inspirant majoritairement de la transposition au domaine de prédiction de trajectoire du mécanisme de *soft-attention* utilisé avec le paradigme récurrent dans le domaine du traitement du langage naturel. Sont présentés principalement, un module d'attention sociale utilisant les trajectoires passées de l'agent principal et de ses voisins pour modéliser leurs interactions ainsi qu'un module d'attention spatiale utilisant l'image de la scène en vue de dessus pour modéliser les interactions de l'agent avec la structure spatiale de la scène.

Dans toutes ces approches, par mimétisme avec le domaine du traitement du langage naturel, on recalcule le module d'attention pour prédire chaque future prédiction. Appliquer par mimétisme ce mécanisme n'est pas nécessairement pertinent. On recalcule les modules d'attention à chaque fois alors que le contexte général ne change pas. Il s'agit toujours de la même scène, ce sont toujours les mêmes agents voisins et leurs mêmes trajectoires observées.

2.5 Les modalités d'évaluation dans les études d'apprentissage profond

2.5.1 Variabilité importante des modalités d'évaluation

La majorité des approches utilisant l'apprentissage profond se mettent d'accord sur l'utilisation de deux métriques :

— ADE (équation 2.16) correspond à la moyenne, pour chaque point d'une sous-trajectoire prédite T_{pred}, de la distance euclidienne (équation 2.18) entre ce point $T^{(i)}_{pred}$ et celui correspondant dans la sous-trajectoire réelle $T^{(i)}_{réelle}$. Elle permet d'évaluer la capacité du modèle à prédire une trajectoire dont la forme est similaire à celle de la trajectoire réelle attendue.

— FDE (équation 2.17) correspond à la distance euclidienne entre le dernier point de la sous-trajectoire prédite et le dernier point de la sous-trajectoire réelle attendue. Cette métrique permet d'évaluer la capacité du modèle à prédire la destination de l'agent.

$$ADE(T_{pred}, T_{réelle}) = \frac{1}{|T_{pred}|} \sum_{i=1}^{|T_{pred}|} L2(T_{pred}^{(i)}, T_{réelle}^{(i)}) \tag{2.16}$$

$$FDE(T_{pred}, T_{réelle}) = L2(T_{pred}^{(|T_{pred}|)}, T_{réelle}^{(|T_{pred}|)}) \tag{2.17}$$

$$L2(X_1, X_2) = \sqrt{(X_1^{(0)} - X_2^{(0)})^2 + (X_1^{(1)} - X_2^{(1)})^2} \tag{2.18}$$

On désigne par $|T_{pred}|$ la longueur de la trajectoire T_{pred}. On désigne par X_i un vecteur à deux dimensions. La première dimension est notée $X_i^{(0)}$ et la deuxième $X_i^{(1)}$. Ces deux métriques évaluent la capacité du modèle à prédire une trajectoire la plus proche possible, en terme de positions, de la trajectoire réelle.

On peut observer sur l'ensemble des études relatives à la prédiction de trajectoires, que les ensembles de données utilisés sont relativement nombreux. Par exemple, Fernando et al. [54] utilisent les ensembles de données *New-York Grand Central* et *Edinburgh Informatics Forum*. Radwan et al. [52] utilisent les ensembles de données *UCY*, *ETH* et introduisent l'ensemble de données *Freiburg Street Crossing*. Ce ne sont que quelques exemples mais jusqu'à un certain point, on trouve plus ou moins un ensemble de données par étude. Plus spécifiquement, pour les approches utilisant l'apprentissage profond, deux ensembles de données semblent avoir été adoptés presque uniformément par la communauté. Il s'agit des ensembles *UCY* et *ETH*. Ces deux ensembles de données regroupent cinq scènes. Ces deux ensembles de données regroupent moins de 2k agents. Tous les agents sont des piétons. Ces ensembles de données regroupent un petit nombre de trajectoires, peu de scènes et uniquement des piétons. Les scènes sont structurées spatialement simplement, ce sont des larges trottoirs où les trajectoires des agents ne sont pas majoritairement contraintes spatialement par la scène. La variété de patrons de déplacement y est pauvre. Principalement, les interactions sociales entre agents sont de l'ordre de l'évitement au dernier instant. Les agents éloignés n'ont a priori pas ou peu d'influences entre eux. A priori, ces ensembles ne mettent pas en scène suffisamment d'interactions avec l'environnement pour permettre d'évaluer l'aptitude d'un modèle à les prendre en compte. Le *SDD* propose une plus grande sélection de scènes parmi lesquelles des structures spatiales plus complexes donnant lieu à des interactions entre agent et espace. Le nombre d'agent y est bien supérieur, environ 20k. Et l'ensemble de données regroupe presque à part égale des piétons et des cyclistes ainsi qu'une part minoritaire d'autres types d'usagers. Cette mixité donne lieu à une plus grande variété d'interactions, entre agents de types différents notamment. Cet ensemble de données est utilisé par Sadeghian et al. dans deux études [6, 56]. Cependant, dans ces études, seul les piétons sont considérés. Cela réduit le nombre et la variété des interactions sociales et spatiales. Aussi, comment prédire

le mouvement d'un piéton qui évite un usager d'un autre type qui n'est pas considéré par le modèle ? Néanmoins, l'ensemble de données utilisé de cette manière offre une meilleure possibilité d'évaluer des modèles plus complexes.

Les études de Becker et al. [63], Nikhil et al. [5] et Sadeghian et al. [6, 56] mettent en avant que sur les ensembles de données simples comme *ETH/UCY*, les modèles complexes sont souvent moins bons que les modèles naïfs. L'utilisation de *SDD*, bien qu'amputé de la moitié de ses interactions permet de montrer qu'un ensemble de données plus complexe permet de mettre en avant l'intérêt de modèles complexes. Il est cependant difficile d'évaluer dans quelle mesure c'est le cas puisque Sadeghian et al. [6, 56] présentent leurs résultats en pixels sur *SDD* et en mètres sur *ETH/UCY*. Les résultats en pixels sont moyennés sur l'ensemble des scènes, malgré le fait que le ratio mètre/pixel ne soit pas constant parmi ces scènes. Il est ainsi difficile de se rendre compte de l'impact des améliorations proposées. Ils montrent néanmoins une division par deux en terme d'*ADE* et *FDE* opérée par le modèle complexe sur *SDD*.

Le protocole de validation est lui aussi disparate dans l'ensemble des études mentionnées dans la revue de littérature. Par protocole de validation, on se place dans le contexte de l'apprentissage machine, où il est nécessaire de proposer un protocole de validation correct afin d'évaluer la capacité d'un modèle à généraliser correctement ses connaissances et à ne pas être victime du sur-apprentissage. La liste suivante résume quatre méthodes de validation que l'on peut retrouver dans les études utilisant l'apprentissage profond mentionnées dans la revue de littérature. Parfois le protocole n'est pas explicitement détaillé.

— Validation croisée avec omission de scène. On entraîne sur un sous-ensemble des scènes de l'ensemble de donnée et on teste sur la ou les scènes restantes. On recommence pour chaque permutation.
— On conserve 80% des données de chaque scène pour l'entraînement et 20% des données de chaque scène pour le test.
— On entraîne sur un sous-ensemble des scènes de l'ensemble de données et on teste sur la ou les scènes restantes.
— Validation croisée avec omission de scène. Puis le modèle est ré-entrainé sur 50% des données de la scène d'évaluation et testé sur les 50 % restant.

On présente dans l'annexe A le tableau A.1 récapitulatif des différentes méthode d'évaluation ainsi que les ensembles de données respectivement utilisés dans un certain nombre d'études d'apprentissage profond. Le tableau cherche à illustrer la variabilité des modalités d'évaluation entre les différentes études, mentionnées dans cette partie de la revue de littérature. Le tableau A.2 présenté en annexe A recense les résultats pour toutes les approches

d'apprentissage profond utilisant les métriques *ADE* et *FDE*.

2.5.2 Des métriques insuffisantes pour évaluer l'intérêt des modèles complexes

Les deux métriques majoritaires utilisées, à savoir *ADE* et *FDE* sont insuffisantes pour évaluer la pertinence de modèles complexes par rapport à des modèles naïfs. Ces deux métriques ont deux caractéristiques principales. Elles sont purement positionelles, c'est-à-dire qu'elles évaluent la capacité du modèle à prédire une trajectoire dont les positions sont les plus proches possible de celles de la trajectoire attendue. Elles ne prennent pas en compte l'environnement de l'agent principal dans l'évaluation de la qualité de la prédiction. Or, étant donné le caractère non déterministe du problème de prédiction, (mentionnée dans la section 2.3), évaluer la capacité du modèle à choisir la bonne possibilité étant donné une infinité n'est pas nécessairement pertinent. Observer si le modèle est capable d'effectuer une prédiction réaliste au regard des environnement social et spatial de l'agent principal semble plus intéressant pour évaluer l'aptitude du modèle à faire usage dudit contexte. En d'autres termes si on admet qu'étant donné un contexte, il existe un certain nombre de possibilités de futur mouvement pour l'agent principal respectant les contraintes de ce dernier, alors il est peut-être plus pertinent d'évaluer si la prédiction effectuée par le modèle appartient à cet ensemble ou non, que de savoir si le modèle est capable de sélectionner le bon élément de cet ensemble.

De plus, comme évoqué dans l'étude de Hug et al. [12], plus le nombre de possibilités de mouvement dans une scène est important et plus le modèle utilisé est capable de représenter la distribution de ces possibilités, plus il a de chance de faire une prédiction ne correspondant pas au mode attendu de la distribution. On peut imaginer que le modèle complexe est capable de représenter l'ensemble de la distribution de mouvements possibles, tandis que le modèle naïf aura plus tendance à se rapprocher d'un modèle à vitesse et accélération constante. Ainsi, sur une scène complexe, si on évalue avec les métriques *ADE* et *FDE*, un modèle complexe sera potentiellement désavantagé par rapport à un modèle naïf, car ces métriques évaluent la capacité du modèle à choisir la bonne possibilité parmi l'ensemble des possibilités autorisées.

2.5.3 Des ensembles de données inadaptés au problème

L'ensemble de données majoritairement utilisés *ETH/UCY* est trop simple. Le nombre de patrons de déplacement qu'il contient est limité. Dans la scène ZARA par exemple, les agents ne se déplacent que dans deux directions. Par ailleurs, les déplacements des agents sont peu ou pas influencés par les interactions avec leur environnement. Ainsi, la capacité d'un modèle capable de modéliser une distribution de patrons de déplacement plus complexe

ne pourra pas être évaluée correctement sur de tels ensembles de données. Ces modèles plus complexes seront même susceptibles de donner de moins bons résultats puisqu'ils essaient de modéliser des interactions n'ayant pas nécessairement d'influences sur les trajectoires des agents. Pour finir, on a pu remarquer dans l'étude de Becker et al. [63] que même sur un ensemble de données comme *SDD*, censé être suffisamment complexe, des modèles naïfs avaient de meilleurs résultats que des modèles plus complexes comme le *social-lstm* de Alahi et al. [47] ou le *social-gan* de Gupta et al. [51]. Cela peut s'expliquer pour plusieurs raisons, les modèles utilisant les interactions sont plus adaptés pour des scènes denses où les agents sont proches les uns des autres et où le module d'interaction est plus utilisé pour évitement à la dernière seconde que pour prise en compte générale à moyen terme des interactions. Ce qui n'est pas le cas pour *SDD* si on considère uniquement les piétons. Par conséquent, ces modèles complexes possèdent des modules inadaptés à un tel ensemble de données en comparaison aux modèles naïfs.

Il est donc nécessaire à la fois d'étudier des ensembles de données proposant des patrons de déplacements variés ainsi que des interactions agent/environnement riches et de proposer de nouvelles métriques adaptées à l'évaluation.

2.5.4 Bilan

Pour conclure, les modalités d'évaluation tant au niveau, des ensembles de données que des métriques utilisées ne sont pas adaptées pour évaluer la qualité des modèles complexes par rapport aux modèles simples. Les modèles d'attention sont utilisés de manière mimétique à ceux utilisés dans le traitement du langage naturel ce qui n'est pas forcement optimal pour la qualité de prédiction ou pour l'efficacité de calcul. Ils sont par défaut employés dans le paradigme récurrent alors que les modèles utilisant le paradigme convolutif sont tout aussi performants et plus rapides.

CHAPITRE 3 MÉCANISMES D'ATTENTION POUR LES MODÈLES CONVOLUTIFS DANS LE CADRE DE LA PRÉDICTION DE TRAJECTOIRES

Le but de cette partie est de présenter la méthodologie utilisée pour étudier l'utilisation des mécanismes d'attention conjointement avec des modèles de prédiction convolutifs. On présente dans un premier temps les modèles de bases nécessaires pour la comparaison avec les modèles proposés. Dans un deuxième temps, on présente les modèles proposés en détaillant leurs architectures, leurs spécificités ainsi que leurs modalités d'entraînement. Dans un troisième temps, on décrit l'ensemble de données utilisé pour les expériences ainsi que les pré-traitements qui lui sont appliqués. Dans un quatrième temps, on revient sur les caractéristiques de la tâche de prédiction et sur les modalités d'évaluation des modèles. Enfin, on présente les métriques utilisées pour l'évaluation.

3.1 Modèles de base

On implémente deux catégories de modèles de base, une catégorie de modèles de base naïfs, ne prenant en compte que la trajectoire partielle passée d'un agent pour prédire sa trajectoire future, et une catégorie de modèles appartenant au paradigme récurrent avec module d'attention.

3.1.1 Les modèles de base naïfs

Le premier modèle de base implémenté est le *LSTM-MLP* de Becker et al. [63]. On le choisit car c'est le modèle naïf récurrent le plus simple et le plus performant. Il est constitué en deux parties, un encodeur *LSTM* pour encoder la trajectoire partielle observée, puis un *MLP* qui prend en entrée cet encodage et prédit simultanément l'ensemble des futures positions de l'agent.

On souhaite aussi utiliser un modèle de base naïf convolutif. On utilise le *CNN-MLP* proposé par Nikhil et al. [5] illustré par la figure 2.10 et décrit dans la section 2.3.

Ces deux modèles de base servent plusieurs buts :

— Disposer d'un modèle de base convolutif et d'un modèle de base récurrent permet de vérifier leur équivalence en terme de performances de prédiction sur l'ensemble de données utilisé dans cette étude.

— Disposer d'un modèle de base convolutif et d'un modèle de base récurrent permet de

vérifier que le modèle convolutif est plus rapide pour effectuer une prédiction que le modèle récurrent.
— Le mécanisme d'encodage convolutif choisi sera aussi utilisé comme bloc de base des modèles attentifs proposés dans la section suivante.
— Disposer de ces modèles de base permet d'étudier l'intérêt d'utiliser un composant d'attention par étude ablative (voir section suivante)

3.1.2 Les modèles de base attentifs

Les modèles de base attentifs proposés reprennent les caractéristiques du modèle de *soft-attention* ou attention visuelle de Xu et al. [23] (voir section 2.1.7). On reprend les caractéristiques de leur modèle car ce sont ces modules d'attention qui sont utilisés pour la tâche de prédiction de trajectoire dans l'approche *sophie-GAN* (Sadeghian et al. [6]), seul modèle utilisant l'attention avec des poids non définis manuellement pour prendre en compte les interactions agent/agent et agent/espace. Ces modèles de base reposent sur l'architecture encodeur/décodeur *LSTM*. On considère deux modèles de base attentifs, un modèle prenant en compte les interactions agent/agent et l'autre les interactions agent/espace.

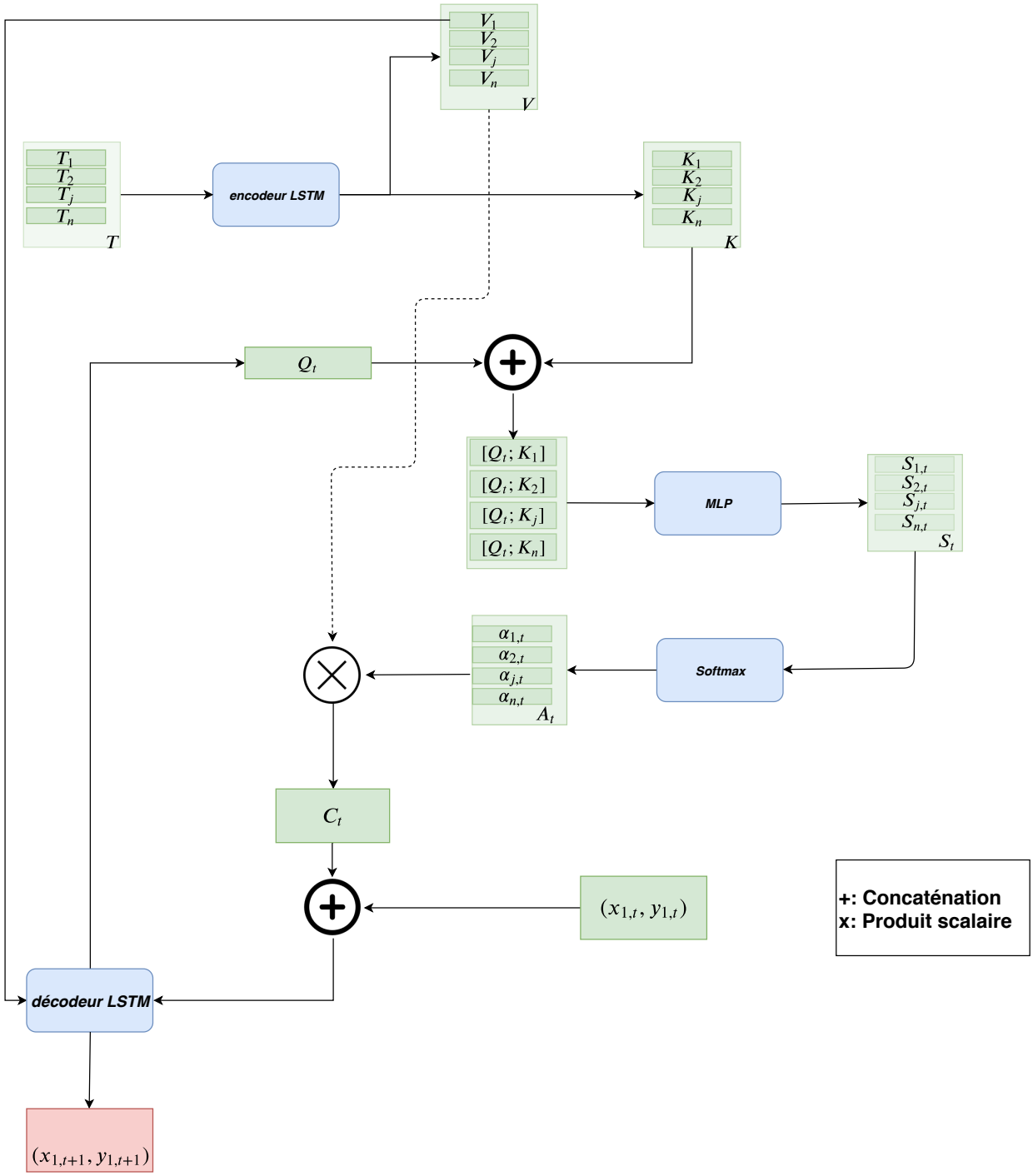


Figure 3.1 Module de soft-attention pour les interactions agent/agent présenté selon le prisme de l'attention généralisée

La figure 3.1 représente le module de *soft-attention* utilisé comme modèle de base attentif prenant en compte les interactions agent/agent. Le modèle est formalisé selon les règles de

l'attention généralisée (section 2.1.7). Les trajectoires observées de l'agent principal T_1 et de ses voisins $\{T_j, \forall\, \mathrm{j} \in [2\,;\mathrm{n}]\}$ sont d'abord encodées par un encodeur *LSTM*. Les agents voisins correspondent à tous les agents présents dans la scène en même temps que l'agent principal. Les encodages obtenus sont utilisés à la fois comme *valeurs* et comme *clés*. On note Q_t la *requête* nécessaire, à l'instant t, pour prédire la future position de l'agent principal à l'instant *t+1*. On associe à Q_t, l'état caché du décodeur *LSTM* obtenu lors de la prédiction de la position précédente. Le score (équations 3.1 et 3.2) entre Q_t et K est calculé en concaténant chacune des clés K_i de K à Q_t et en projetant chacun des vecteurs obtenus dans un espace à une dimension avec un *MLP*. On obtient ainsi un score $S_{i,t}$ par paire K_i/Q_t, regroupés dans le vecteur de score S_t. On applique la fonction *softmax* sur le vecteur de score pour obtenir un vecteur de poids A_t sommant à un (équation 3.3). On applique le produit scalaire entre V et S_t pour obtenir le vecteur d'attention C_t (équation 3.4). Optionnellement, on concatène à C_t la dernière position prédite de l'agent principal ($(x_{1,t}, y_{1,t})$) (Si il s'agit de la première prédiction, on prend la dernière position de la trajectoire observée). Le résultat de l'opération précédente est pris en entrée par le décodeur pour prédire la prochaine position ($(x_{1,t+1}, y_{1,t+1})$) (équation 3.5). L'état caché du décodeur est initialisé pour la première prédiction avec l'encodage de la trajectoire observée de l'agent principal V_1. Le processus est répété de manière récurrente pour la prédiction de chaque position.

$$S_{i,t} = score(K_i, Q_t) = MLP([K_i; Q_t]) \tag{3.1}$$

$$S_t = score(K, Q_t) \tag{3.2}$$

$$A_t = softmax(S_t, Q_t) \tag{3.3}$$

$$C_t = A_t^T \cdot V \tag{3.4}$$

$$(x_{1,t+1}, y_{1,t+1}) = LSTM_{dec}([C_t, (x_{1,t}, y_{1,t}), V_1]; Q_t) \tag{3.5}$$

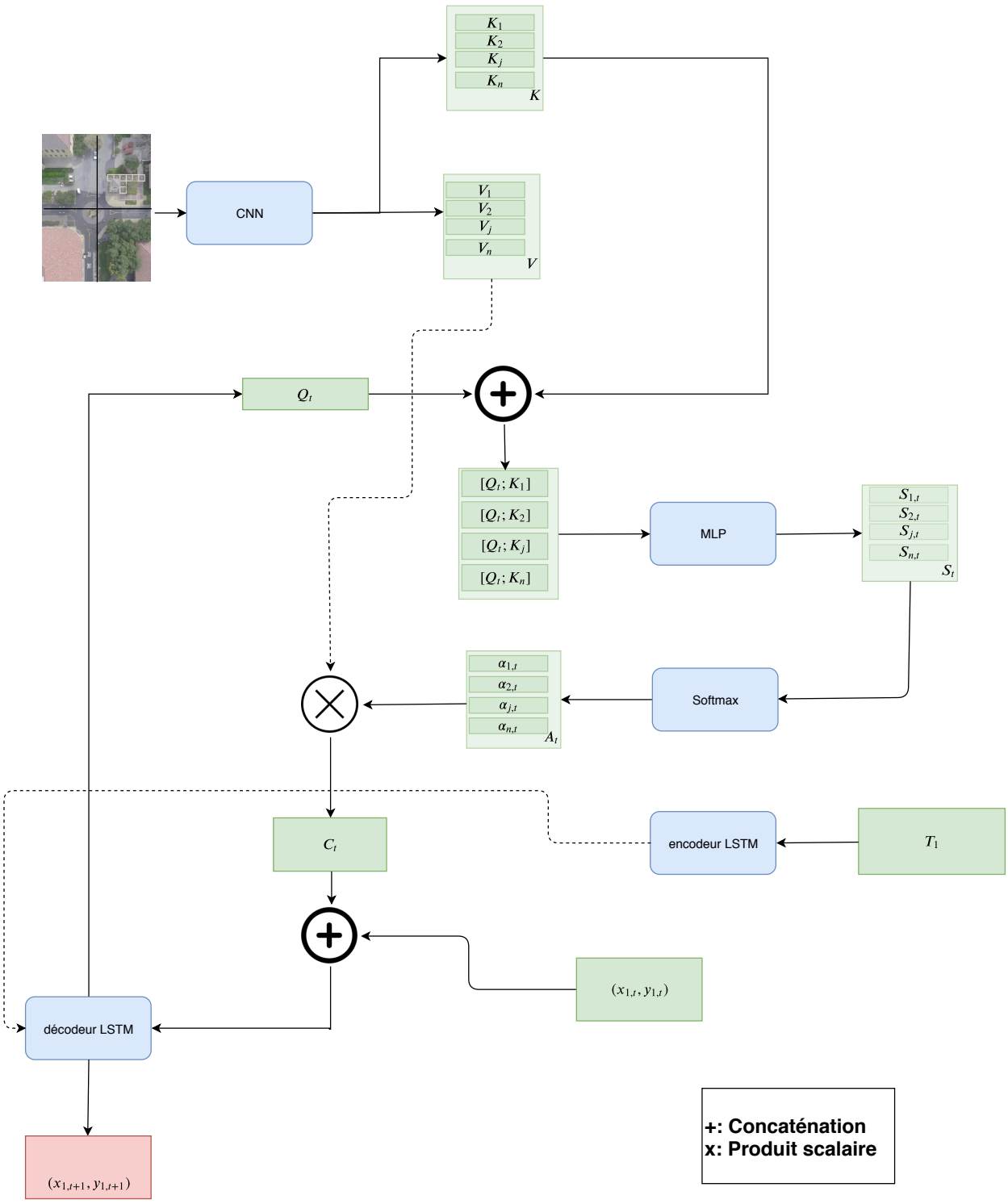


Figure 3.2 Module de soft-attention pour les interactions agent/espace présenté selon le prisme de l'attention généralisée

Le modèle de base attentif utilisé pour représenter les interactions spatiales (figure 3.2), est similaire au précédent, à ceci prêt que l'attention n'est plus utilisée sur les trajectoires observées des voisins mais sur l'image de la scène. Deux différence sont à noter :

— On encode uniquement la trajectoire observée de l'agent principal T_1 et non plus celles de ses voisins.

— Les *clés* et *valeurs* correspondent aux vecteurs d'attributs obtenu à partir d'un *CNN* pré-entraîné appliqué sur l'image de la scène, comme dans Sadeghian et al. [6, 56]. On rappelle que chaque vecteur d'attributs caractérise une sous-partie de l'image.

Il est important de noter que pour les deux modèles avec attention, le vecteur d'attention est recalculé pour la prédiction de chacune des futures positions. D'un point de vue sémantique, l'état caché du décodeur à l'instant t, utilisé comme la requête Q_t représente le contexte de prédiction à l'instant t. En effet, il conserve partiellement en mémoire les précédentes prédictions effectuées, l'encodage de la trajectoire observée de l'agent principal ainsi que les précédents résultats du module d'attention. Un poids du module d'attention sociale correspond à l'importance à donner à la trajectoire passée d'un agent voisin étant donné le contexte courant de prédiction. Un poids du module d'attention spatiale correspond à l'importance à donner à une partie de l'image de la scène étant donné le contexte courant de prédiction.

Les deux modèles de base n'ont pas été présentés tels quels dans la littérature. Ils sont cependant fortement inspirés des approches de Sadeghian et al. [6, 56] et de Fernando et al. [54]. Pour faire simple, on a extrait les modules d'attention des approches plus complexes proposées dans leurs études.

Ces deux modèles de base revêtent plusieurs utilités :

— Evaluer et confirmer l'impact des modules d'attention spatiaux et sociaux traditionnels (présentés dans *sophie-GAN* (Sadeghian et al. [6])) par rapport au modèle de base récurrent naïf.

— L'utilisation des modules d'attention dans le cadre des modèles convolutifs entraine des modifications dans les architectures d'attention utilisées qui conduisent à une variation de la portée sémantique de ces architectures. Ainsi, les deux modèles de base permettront d'évaluer l'impact de ce changement de sémantique en terme de qualité de prédiction et d'évaluer l'impact du changement d'architecture sur le temps de calcul nécessaire pour faire une prédiction.

3.2 Les modèles proposés

Les modèles proposés reposent tous sur la même structure. À la place de l'encodeur *LSTM*, on utilise le *CNN* de Nikhil et al. [5] pour encoder les trajectoires observées. On dispose de deux structures distinctes, une pour l'attention sociale et une pour l'attention spatiale. Il est important de noter que la transition entre le paradigme récurrent et le paradigme convolutif impose de nouvelles contraintes sur les mécanismes d'attention. En effet, dans les architectures basées sur le modèle encodeur/décodeur *LSTM*, l'état caché du décodeur sert de requête dans les modules d'attention. En effectuant la transition vers des modèles non récurrents, on perd la possibilité d'utiliser cet état caché comme requête. Le fait de ne pas avoir de décodeur oblige à utiliser un autre type de requête.

Pour rappel, dans les modèles de base proposés dans la sous-section 3.1.2, la requête évolue pour chaque prédiction. En d'autres termes, le vecteur d'attention est réévalué pour prédire chacune des futures positions de l'agent. De cette manière, pour associer une valeur de pertinence à tous les objets constituant l'environnement d'un agent lors de la prédiction de sa future position, on considère à la fois la trajectoire de l'agent pendant la période d'observation et les précédentes positions prédites pour cet agent. Cette manière de faire conduirait selon nous à plusieurs écueils :

- Réaliser les prédictions de manière récurrente, conduit à une augmentation progressive de l'erreur au fur-et-à-mesure que l'on avance dans l'horizon de prédiction comme mentionné dans Becker et al. [63].
- Mettre à jour le contexte social de prédiction pour chaque future position pourrait être inutile. En effet, dans le module d'attention, on considère pour les agents voisins uniquement leurs trajectoires pendant la période d'observation. Mais quand on prédit la future position de l'agent principal, on ne prend pas en compte l'évolution des trajectoires des agents voisins. Autrement dit, les trajectoires des agents voisins sont figées pour la période d'observation et recalculer le contexte pour prédire chacune des futures positions de l'agent principal pourrait s'avérer redondant puisque le contexte ne change pas au fil de la prédiction, et ainsi le gain d'information apportée en recalculant l'attention à chaque prédiction pourrait s'avérer nulle.
- Mettre à jour le contexte spatial de prédiction pour chaque future position serait peut-être plus utile que pour le contexte social. En effet, le contexte spatial ne dépend pas du temps. Recalculer un module d'attention spatiale pour chaque prédiction permettrait, par exemple, d'éviter un obstacle à chaque instant de prédiction. Néanmoins, calculer le module d'attention pour chaque prédiction est coûteux en temps de calcul et pas forcément pertinent surtout si l'intervalle de temps entre deux positions est faible

(inférieur à une seconde par exemple).
Les deux derniers points seront évalués par les expériences proposées dans ce projet.

La solution proposée et évaluée dans ce travail, fait l'hypothèse suivante. Pour prédire toutes les futures positions de l'agent principal simultanément sur l'ensemble de l'horizon de prédiction, le seul élément nécessaire pour associer une valeur de pertinence à tous les objets constituant l'environnement (ou requête), est la trajectoire passée de l'agent principal. On considère que l'agent prend une décision à la fin de la période d'observation pour l'ensemble de sa trajectoire pendant la période de prédiction. On prédit ainsi toutes les futures positions simultanément pour réduire l'accumulation de l'erreur, on observe les contextes social et spatial une unique fois pour prédire l'ensemble de la future trajectoire, réduisant ainsi le temps de calcul nécessaire à la prédiction. Enfin, pour adresser le fait que recalculer le contexte spatial à chaque instant de prédiction pourrait être pertinent, on propose d'utiliser le mécanisme d'attention multi-tête de Vaswani et al. [4] pour permettre au modèle de prédiction de focaliser son attention sur plusieurs combinaisons des objets constituant l'environnement au moment d'effectuer la prédiction.

3.2.1 Modèle proposé pour prendre en compte les interactions sociales

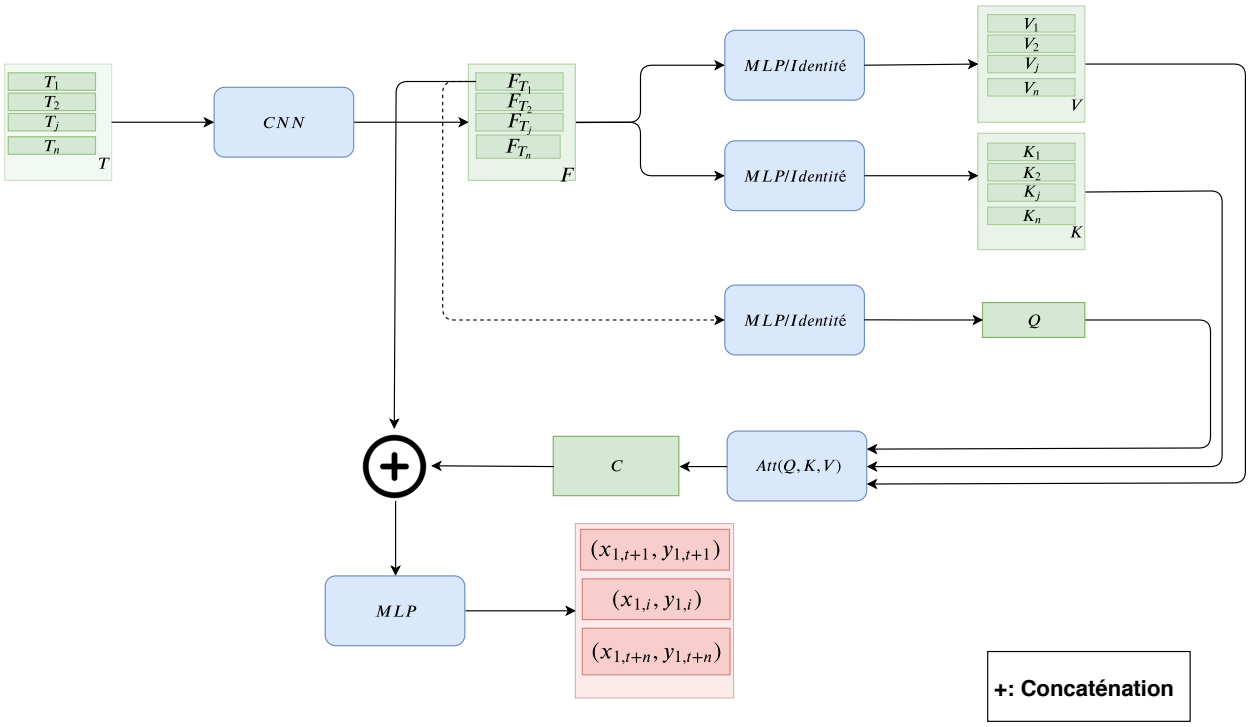


Figure 3.3 Patron général pour les modèles convolutifs avec attention sociale

La figure 3.3 illustre le modèle de prédiction proposé pour prendre en compte les interactions agent/agent lors de la prédiction. Le modèle est formalisé selon les règles de l'attention généralisée 2.1.7. Les trajectoires observées de l'agent principal T_1 et de ses voisins $\{T_j, \forall$ j $\in [2;n]\}$ sont d'abord encodées par le *CNN* de Nikhil et al. [5]. On obtient un vecteur d'attributs F_{T_j} par trajectoire observée. Pour obtenir les *clés*(K) et *valeurs*(V), deux options sont possibles. Soit on prend directement les vecteurs d'attribut F_{T_j} à la fois comme *clés* et *valeurs*, soit on projette les vecteurs d'attribut avec un *MLP* pour les *clés* et un *MLP* pour les *valeurs* respectivement (pour rappel, cela permet d'associer des informations différentes pour les *clés* et les *valeurs*. Pour obtenir, la *requête*(Q), on prend le vecteur d'attributs de l'agent principal que l'on projette ou non avec un *MLP*. Les *clés*, *valeurs* et *requête* sont passées dans le module d'attention $Att(Q, K, V)$ pour obtenir le vecteur d'attention C. Le vecteur C est concaténé au vecteur d'attribut de l'agent principal F_{T_1}. Le résultat de la concaténation est passé dans un *MLP* pour prédire l'ensemble des futures positions simultanément. Différents modèles d'attention pourront être utilisé pour $Att(Q, K, V)$. La projection des *clés*, *valeurs* et *requête* est laissée en option puisqu'elle dépend du mécanisme de *soft-attention* utilisé.

3.2.2 Modèle proposé pour prendre en compte les interactions spatiales

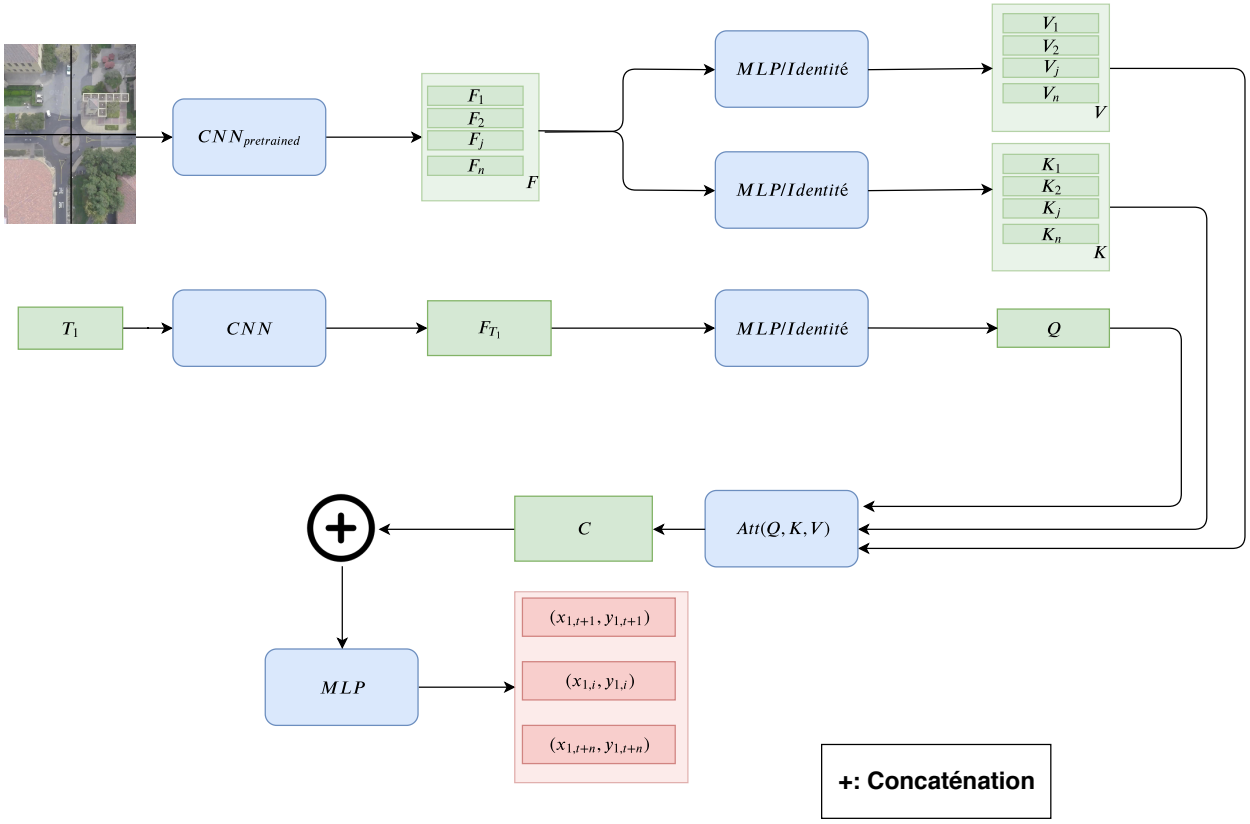


Figure 3.4 Patron général pour les modèles convolutifs avec attention spatiale

La figure 3.4 illustre le modèle de prédiction proposé pour prendre en compte les interactions spatiales lors de la prédiction. Il fonctionne de manière similaire au modèle proposé pour l'attention sociale à l'exception des *valeurs* utilisées dans le module d'attention. La trajectoire observée de l'agent principal T_1 est encodée (F_{T_1}) par le *CNN* de Nikhil et al. [5] puis utilisée comme *requête*. Comme dans Sadeghian et al. [6, 56], l'image de la scène est passée dans un *CNN* pré-entrainé pour extraire un vecteur d'attributs F_j par partie de l'image. Ces vecteurs d'attributs seront utilisés comme *valeurs* et *clés* dans ce modèle. Comme dans le modèle précédent, la projection des *clés*, *valeurs* et *requête* est laissée en option et dépendra du mécanisme de *soft-attention* utilisé.

3.2.3 Mécanisme de soft-attention

Les mécanismes d'attention proposés dans cette section peuvent être insérés dans les deux modèles précédents, à la place du bloc $Att(Q, K, V)$. Pour chacun des deux modèles

étudiés, on propose deux mécanismes d'attention différents. Le premier est le mécanisme de *soft-attention* utilisé dans les modèles de base attentifs de la section 3.1.2. Le deuxième est le mécanisme d'attention multi-tête développé dans la section 2.1.7.

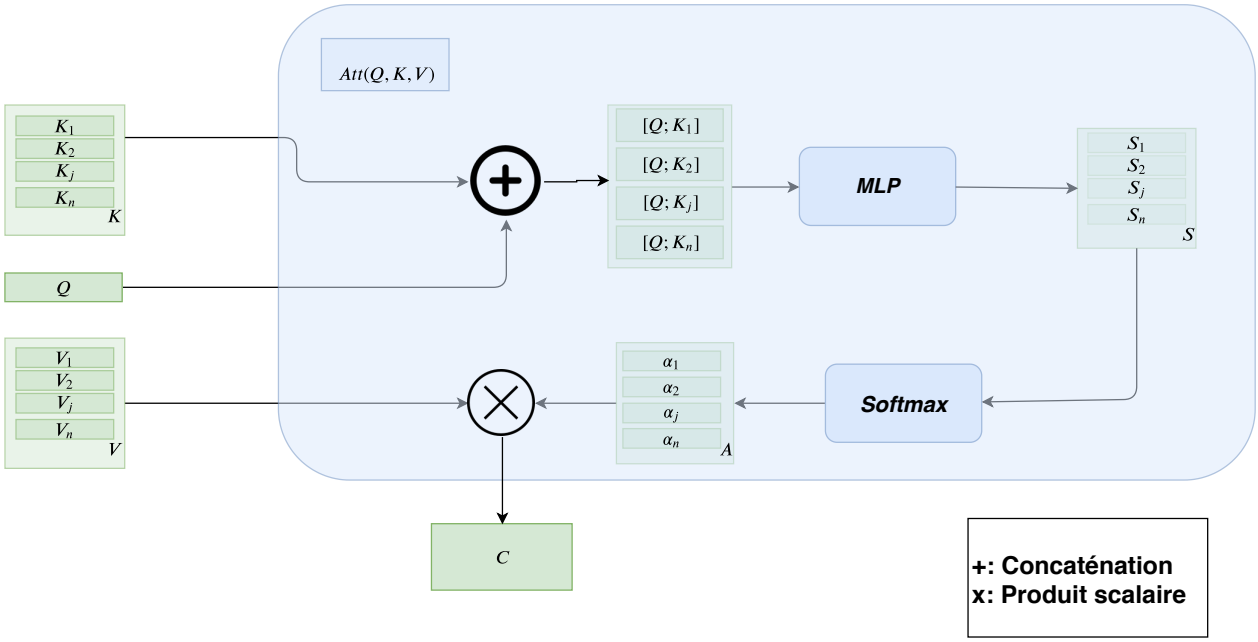


Figure 3.5 Module de soft-attention

La figure 3.5 illustre le mécanisme de *soft-attention* sous le formalisme de l'attention généralisée. Dans le cadre du mécanisme de *soft-attention*, les *clés*, *valeurs* et *requêtes* sont obtenues sans projection. En d'autres termes elles prennent directement les vecteurs d'attributs sans les passer dans un *MLP*. Chaque paire de *clés* et de *requête* sont concaténées ensembles. Chaque vecteur issu de la concaténation est projeté dans un espace a une dimension par un *MLP* nous donnant le score associé à la paire correspondante. Les scores sont normalisés par une fonction *softmax*. La somme des *valeurs* pondérées par leurs poids respectifs constitue le vecteur d'attention C.

3.2.4 Mécanisme d'attention multi-tête

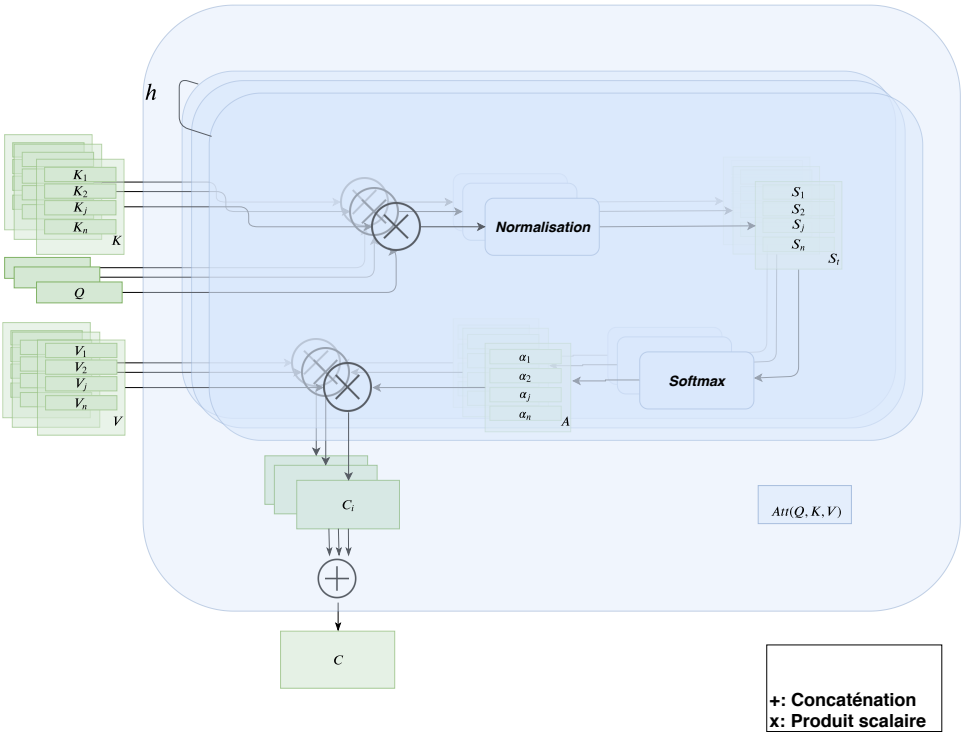


Figure 3.6 Module d'attention multi-tête

La figure 3.6 illustre le mécanisme d'attention multi-tête sous le formalisme de l'attention généralisée. Dans le cadre du mécanisme d'attention multi-tête, les *clés*, *valeurs* et *requêtes* sont obtenues avec projection. En d'autres termes elles sont projetées chacune dans un espace différent par un *MLP*. Les trois espaces de projection ont la même dimension p (p est un hyper-paramètre du modèle). Ici, la projection est utilisée avant le module d'attention car celui-ci ne contient pas de poids à apprendre par rétro-propagation. Le modèle doit ainsi apprendre à projeter les *clés*, *valeurs* et *requêtes* dans des espaces qui seront mis en relation par produit scalaire permettant d'obtenir les poids à associer à chaque valeur. Chaque paire de *clés* et de *requête* sont multipliées ensembles par produit scalaire. Le produit scalaire est divisé par $\sqrt{p}$ dans le bloc *Normalisation* de la figure. Le résultat de chaque produit scalaire est un réel correspondant au score associé à la paire *clé/requête*. Les scores sont normalisés par une fonction *softmax*. La somme des valeurs pondérées par leurs poids respectifs constitue le vecteur d'attention C_i de la tête d'attention i . Le processus est répété en parallèle pour les h têtes d'attention. Les vecteurs d'attention de chaque tête C_i sont concaténés ensemble pour former le vecteur final d'attention C. L'intérêt principal du module d'attention multi-tête est sa rapidité de calcul. On effectue une projection suivie d'un produit scalaire soit deux

opérations très rapides et facilement parallélisables.

Dans cette partie, on a présenté les modèles pour prédire la trajectoire d'un seul agent à la fois. On peut aussi prédire les trajectoires de tous les agents présents dans la scène pendant la période d'observation de manière simultanée. Cela nécessite de calculer un vecteur de contexte d'attention par agent présent dans la scène. Dans ce cas là, Q contient plusieurs requêtes. C'est dans ce cas que le produit scalaire est particulièrement avantageux puisque une simple multiplication entre la matrice des requêtes Q et celle des clés K permet de calculer l'ensemble des scores nécessaires pour le module d'attention de chacun des agents.

3.2.5 Intérêt des modèles proposés

En proposant les deux architectures non récurrentes (sociale et spatiale) et les deux mécanismes d'attention (*soft-attention* et attention multi-tête), on cherche à évaluer les points suivants :

- On compare les modèles de base attentifs avec les architectures non-récurrentes proposées. Pour les architectures non-récurrentes, on utilise le module de *soft-attention* et celui d'attention multi-tête avec une seule tête. On peut ainsi étudier l'impact du changement de sémantique induit, sur les modules d'attention, par les contraintes inhérentes aux architectures convolutives.
- On compare pour les modèles proposés les performances du module de *soft-attention* et d'attention multi-tête avec une seule tête pour évaluer leur impact sur la qualité et la rapidité de prédiction.
- On étudie l'impact de l'utilisation de plusieurs têtes d'attention pour les interactions agent/agent et agent/espace sur la qualité et la rapidité de prédiction.
- On étudie l'impact de faire les prédictions pour tous les agents présents dans la scène de manière conjointe ou non.

3.3 L'ensemble de données utilisé

Les expériences seront conduites sur l'ensemble de données *Stanford Drone Dataset* de Robicquet et al. [45]. Le *SDD* regroupe les trajectoires issues de six moyens de locomotion différents : des piétons, des vélos, des voitures, des skate-boards, des bus et des charriots. Les deux types d'usagers majoritaires, au moins 95 %, sont les piétons et les cyclistes. Les trajectoires sont extraites à partir de vidéos filmées par drone sur le campus de Stanford. Pour chaque scène, on dispose d'une image en vue de dessus et les coordonnées des trajectoires sont exprimées en pixels dans le référentiel de l'image de la scène. Cet ensemble de données

présente de surcroit les particularités suivantes :

— La variété des scènes proposées est importante (figure 3.7). Il y a huit localisations différentes, et pour chacune d'entre elles plusieurs cadrages. Les structures spatiales des scènes sont elles-aussi diverses. On retrouve des structures plus simples comme des zones piétonnes ainsi que des structures plus complexes comme des intersections de routes, des trottoirs ou encore des carrefours giratoires. Cette diversité permet de complexifier le problème de prédiction à moyen terme qui peut-être trop simple sur des scènes où tous les agents ont tendance à se diriger dans la même direction (cf. ensemble de données ETH/UCY).

— Le fait d'avoir deux types d'usagers différents en proportions équivalentes permet d'étudier les interactions entre des usagers parcourant les scènes de différentes manières et permet de traiter des situations intéressantes/assez complexes nécessitant la prise en compte des interactions sociales.

— Le nombre d'usagers présents sur la totalité de l'ensemble de données est de presque 20k contre environ 2k pour ETH/UCY. La quantité importante de données permet d'entraîner des modèles plus complexes et de disposer d'exemples plus variés d'interactions.

Toutes les études mentionnées dans la revue de littérature ne prennent en compte que les trajectoires des piétons contrairement à la notre où tous les types d'usagers seront utilisés.

Figure 3.7 Un exemple de la variété des structures spatiales proposées par l'ensemble de données

Pour évaluer la performance finale de nos modèles, on sépare l'ensemble de données en deux parties, un ensemble d'entraînement et un ensemble de test. En pratique, on met de côté, pour l'ensemble de test, un sous-ensemble de scènes qui ne seront utilisées que pour

évaluer la performance finale de nos modèles. Les scènes mises de côté sont soit inédites (absente de l'ensemble d'entraînement) soit présentes dans l'ensemble d'entraînement mais sous un cadrage (point de vue) différent. La construction d'un tel ensemble de test permet d'évaluer la capacité des modèles à généraliser sur de nouvelles scènes ou des scènes provenant d'une distribution similaire à celle de l'ensemble d'entraînement. On utilise le découpage du challenge *trajnet* de Sadeghian et al. [13].

Pour sélectionner les modèles et régler leurs hyper-paramètres, on ne peut pas utiliser l'ensemble de test, sinon on risquerait le sur-apprentissage de cet ensemble. Pour résoudre ce problème, on utilise traditionnellement la validation croisée ou une de ses différentes variantes. On peut découper l'ensemble de données en deux parties aléatoirement choisies (souvent selon les proportions 80/20 % pour l'entraînement et le test respectivement). On peut diviser l'ensemble de données en k sous-parties, entraîner sur *k-1* sous-parties et évaluer sur la partie restante et recommencer pour chaque permutation.

La deuxième méthode, découpant l'ensemble d'entraînement en k sous-ensembles, est préférable en général puisqu'elle permet une meilleure approximation de l'erreur de généralisation. Dans notre cas, l'ensemble de données utilisé est trop grand pour répéter plusieurs fois le processus entraînement/validation. On propose un ensemble de validation sur le même modèle que celui de test (c'est-à-dire que l'on met de côté un sous-ensemble des scènes de l'ensemble d'entraînement pour constituer l'ensemble d'évaluation). On met de côté un sous-ensemble des scènes de l'ensemble d'entraînement pour la validation. On choisit les scènes pour qu'elles couvrent une variété suffisante de structures spatiales.

Une fois que les modèles ont été choisis et affinés sur l'ensemble de validation, on ré-entraîne les modèles sur les ensembles de validation et d'entraînement conjointement puis on les teste sur l'ensemble de test.

3.4 Le prétraitement des données et la création des exemples d'entraînement

3.4.1 Prétraitement des données

On effectue plusieurs pré-traitements sur les données avant de les utiliser :

— D'abord, il faut réduire la fréquence d'échantillonnage des trajectoires. Celle-ci est de trente images par seconde. On la réduit à deux-et-demi images par seconde comme dans la majorité des études de la section 2.2.2. On ré-échantillonne en appliquant une spline cubique sur chaque trajectoire.

— Ensuite, à partir de mesures en mètres réalisées manuellement sur une application de cartographie, on effectue la correspondance entre les systèmes de coordonnées des

images et ceux en mètres du monde réel. À partir de ces correspondances, on convertit les coordonnées de l'espace des pixels à l'espace métrique en deux dimensions, afin d'unifier l'unité des résultats entre les différentes scènes.

— Pour réduire l'espace mémoire nécessaire au stockage des données, on réduit le nombre de décimales pour chaque coordonnée. On ne conserve pas plus de deux décimales ce qui correspond à un niveau de précision de l'ordre du centimètre, largement suffisant dans le cadre de notre application.

— Enfin, on élimine les trajectoires des agents dont la première et la dernière position sont dans un cercle de deux mètres de diamètre. Ces agents stationnaires sont soit des erreurs d'étiquetages soit des agents restant statiques tout au long de l'observation d'une scène. Dans tous les cas, ils ne présentent pas nécessairement de valeur ajoutée pour la tâche de prédiction. On choisit donc de les éliminer. On fait ainsi le choix d'ignorer le cas d'un piéton immobile pendant toute la durée d'une scène malgré l'influence qu'il pourrait avoir sur d'autres agents tentant de l'éviter par exemple.

3.4.2 Préparation des exemples d'entraînement

Une fois les données nettoyées, il faut préparer les données pour l'entraînement des modèles de prédiction. Un exemple d'entraînement est créé de la manière suivante : A partir de la trajectoire d'un agent principal, on extrait l'ensemble des séries de vingt unités de temps consécutives. Ce sont les sous-trajectoires. Pour chaque sous-trajectoire, on dispose en plus, des sous-trajectoires des agents voisins présents dans le même intervalle de temps. La sous-trajectoire de l'agent principal est divisée en deux parties : la sous-trajectoire observée, soit les huit premières positions de la sous-trajectoire, et la sous-trajectoire à prédire, soit les douze positions restantes de la sous-trajectoire. Ce découpage est le même que celui du challenge trajnet proposé par Sadeghian et al. [13]. Dans un exemple d'entraînement, on conserve les trajectoires des agents voisins uniquement si elles commencent avant la fin de la sous-trajectoire d'observation de l'agent principal et si elles finissent après le début de la sous-trajectoire à prédire de l'agent principal. La raison étant que si un agent voisin sort de la scène avant la fin de la période d'observation ou qu'il rentre dans la scène après la fin de la période d'observation de l'agent principal, alors il n'a pas d'influence sur la décision de l'agent principal.

Pour gérer la différence de longueur entre les sous-trajectoires des agents voisins (illustrée sur la figure 3.8) et que les correspondances temporelles soient conservées entre les différentes sous-trajectoires, on ajoute des vecteurs de zéros sur chacune des deux dimensions spatiales (x et y) au début et à la fin des sous-trajectoires de longueur inférieure à vingt. Dans le cadre

de la convolution, ajouter des zéros n'a pas d'influence sur le résultat de l'application du filtre de convolution. En outre, le point de coordonnées $(0, 0)$ correspond exactement au coin de chacune des scènes, donc pas un point beaucoup traversé par les agents en règle général, limitant la confusion possible entre le rembourrage et un point de l'espace.

Un exemple d'entraînement est donc indexé sur la trajectoire de l'agent principal et il y a un exemple d'entraînement par nombre de sous-trajectoire de vingt unités de temps contenues dans une trajectoire. Pour réduire la redondance des données d'entraînement, on applique un décalage de trois unités de temps entre chaque sous-trajectoire d'une même trajectoire.

Pour les modèles ne prenant pas en compte les interactions sociales, on ne conserve de l'exemple que la trajectoire de l'agent principal. Pour les modèles prenant en compte les interactions spatiales, on ajoute à l'exemple l'image de la scène en vue de dessus.

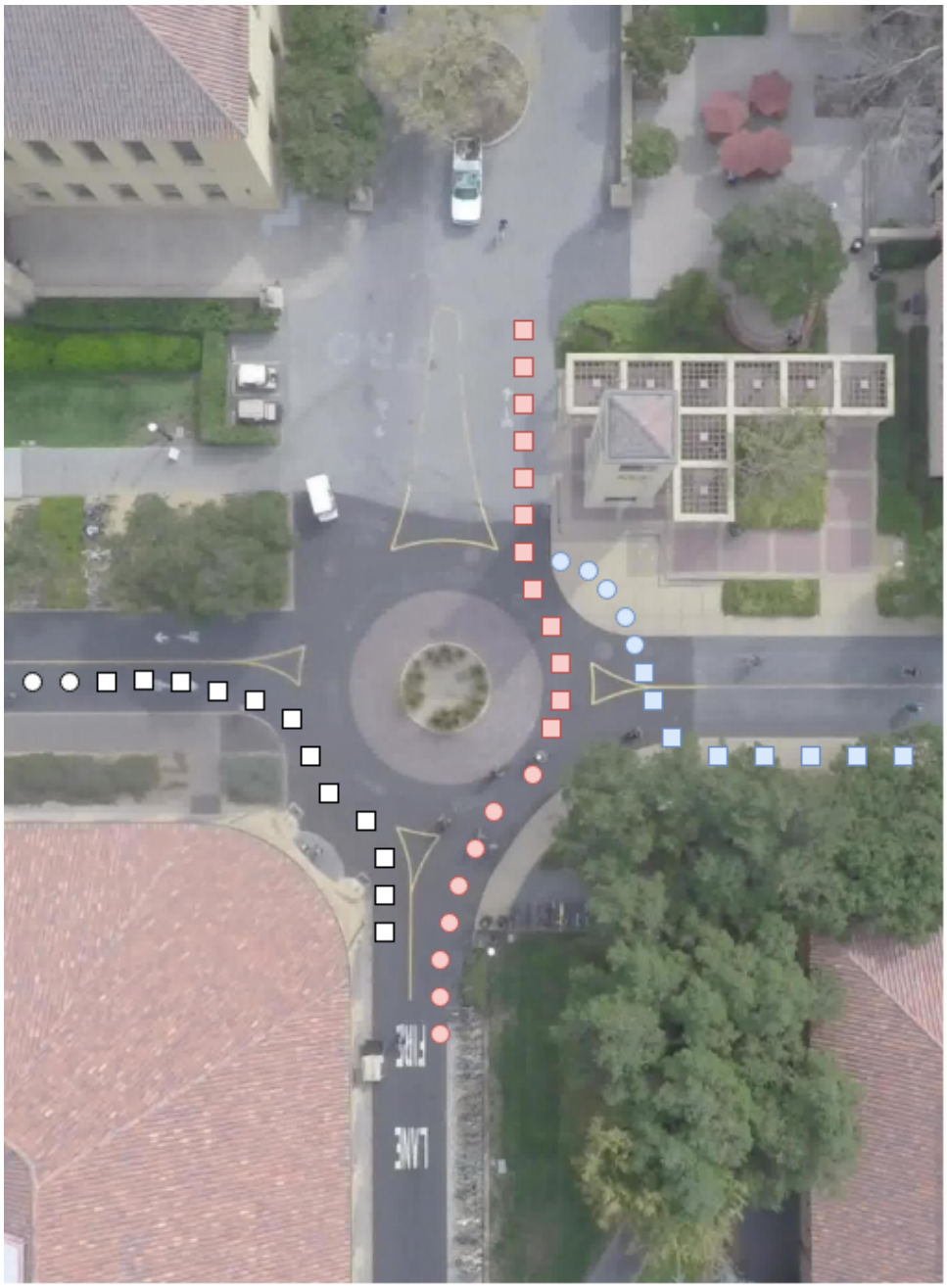


Figure 3.8 Représentation d'un exemple d'entraînement. La trajectoire de l'agent principal est en rouge. Chaque couleur correspond à un agent. Les points et carrés d'une couleur donnée correspondent aux positions successives d'un agent. Les points observés sont représentés par des cercles et les points à prédire sont représentés par des carrés. On peut remarquer que dans ce cas là, seule la trajectoire de l'agent principal est constituée de vingt points contrairement aux deux autres trajectoires dont les longueurs sont inférieures.

3.5 Les modalités d'entraînement

3.5.1 Entrées et sorties

Becker et al. [63] laissent entendre que prédire les positions relatives plutôt que les positions absolues ainsi que prendre en entrée une suite de positions relatives permet de diminuer la complexité de la distribution à modéliser pour le modèle. En effet, l'ensemble des déplacements possibles est relativement restreint comparé à l'ensemble des positions possibles, et ce, sur différentes scènes.

Ainsi, pour représenter les trajectoires en entrée, on propose deux possibilités. La première, utiliser une séquence de positions absolues dans l'espace métrique en deux dimensions. La deuxième, utiliser une séquence de différences de position dans l'espace métrique en deux dimensions. La différence étant toujours exprimée par rapport au point précédent (à gauche dans la figure 3.9).

Pour représenter les trajectoires en sortie, on propose deux possibilités aussi. La première, utiliser une séquence de positions absolues dans l'espace métrique en deux dimensions. La deuxième, utiliser une séquence de différences de position dans l'espace métrique en deux dimensions. La différence étant toujours exprimée par rapport à la dernière position de la sous-trajectoire observée de l'agent (à droite dans la figure 3.9).

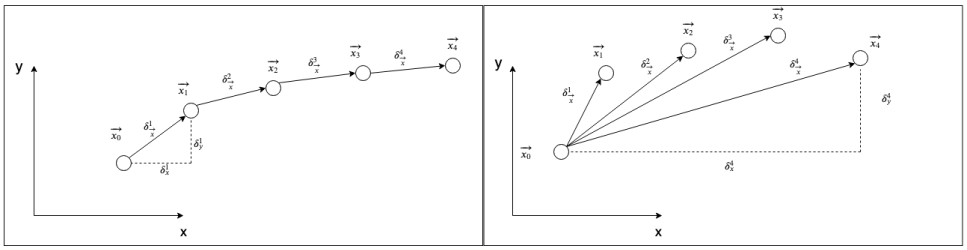


Figure 3.9 Illustration des deux possibilités de représentation de trajectoires comme positions relatives.

Pour chacune des deux options en entrée on propose un mécanisme de normalisation.

— Pour les positions relatives, on applique une standardisation (équation 3.6) (soustraction de la moyenne μ puis division par l'écart-type σ). La moyenne et l'écart-type des différences de positions sont préalablement calculés sur l'ensemble des trajectoires des données d'entraînement.

— Pour les positions absolues, on applique une normalisation *min-max*. Le minimum x_{min} et le maximum x_{max} sont calculés sur toutes les scènes et sur les deux dimensions spatiales confondues. On les calcule de cette manière de telle sorte que les proportions

spatiales dans la représentation normalisée soient conservées d'une scène à l'autre et que l'on ne compresse ou ne dilate pas une dimension spatiale par rapport à l'autre.

$$x_{standardisé} = \frac{x - \mu}{\sigma} \tag{3.6}$$

$$x_{minmax} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.7}$$

3.5.2 Méthode d'entraînement

Pour les modèles de base naïfs, on teste chaque paire de possibilité de format entrée/sortie. À l'exception de celle prenant en entrée des positions relatives et prédisant des positions absolues. Pour les autres modèles, prenant en compte les interactions, le modèle a besoin d'avoir les positions absolues en entrée pour pouvoir mettre en relation spatiale les différents agents entre eux ou avec leur environnement. On se laissera, en revanche, la possibilité de prédire des positions absolues ou relatives.

Les modèles sont entraînés selon l'algorithme de la rétro-propagation du gradient par rapport à une fonction de perte. Comme déjà mentionné, pour les modèles prenant en compte les interactions agent/agent, on peut choisir de prédire uniquement la trajectoire future de l'agent principal en prenant en compte via un mécanisme d'attention les trajectoires des agents voisins. On appellera cette méthode d'apprentissage, *rétro-propagation disjointe*. On peut aussi prédire simultanément les trajectoires de l'agent principal et de ses voisins. On parlera alors de *rétro-propagation conjointe*, étant donné que la fonction de perte sera calculée pour chacun des agents présents dans l'exemple d'entraînement. L'intérêt supposé de cette méthode serait d'encourager le modèle à considérer conjointement les interactions entre agents pour faire des prédictions et de lui permettre d'être entraîné à prédire des trajectoires de tailles variables en observant des trajectoires de tailles variables (mais de taille maximale de huit unités de temps).

Dans le cas de la rétro-propagation conjointe, quand les sous-trajectoires à prédire des agents voisins sont de longueurs inférieures à celle de l'agent principal, on applique un masque pour ne prendre en compte qu'un nombre de prédictions équivalent à la longueur de la sous-trajectoire à prédire réelle. A l'aide du masque, on ignore toutes les prédictions excédentaires lors du calcul de la fonction de perte et de la rétro-propagation du gradient.

Plusieurs variantes de la descente de gradient existent :

— Dans la descente de gradient stochastique, un unique exemple d'entraînement à la fois est fourni au modèle. La fonction de perte est calculée pour un seul exemple

et les poids du modèle sont mis-à-jour étant donné les gradients d'un seul exemple d'entraînement.

— Dans la descente de gradient par lot, le modèle voit la totalité des exemples d'entraînement et accumule leurs gradients respectifs, puis la mise-à-jour des poids du modèle est effectuée pour l'ensemble des gradients.

— Dans la descente de gradient par mini-lot, le modèle utilise un sous-ensemble des exemples d'entraînements à la fois pour mettre à jour ses poids.

Dans ce travail, on utilise la descente de gradient par mini-lot qui permet une meilleure convergence en comparaison à la descente de gradient par lot ainsi qu'une meilleure vitesse d'entraînement en comparaison à une descente de gradient stochastique. L'ensemble d'entraînement est divisé en mini-lot. Quand le modèle a été entraîné sur l'ensemble des mini-lots, on dit qu'il a effectué une itération.

Comme fonction de perte L_{lot} (équation 3.10) entre une trajectoire prédite $T_{j,pred}$ et la trajectoire réelle $T_{j,réelle}$, on calcule l'erreur quadratique (équation 3.8) sur chacune des deux dimensions spatiales. On somme ces erreurs pour l'ensemble des points $T^{(i)}_{j,réelle}$ des deux trajectoires, toutes dimensions confondues, et on obtient un nombre réel $L_{trajectoire}(T_{j,réelle}, T_{j,pred})$ (équation 3.9). Pour plusieurs trajectoires, on fera la moyenne de la valeur obtenue à l'étape précédente (équation 3.10). On note $T^{(i)}_{x_{j,réelle}}$ la valeur pour la coordonnée x du *i-ème* point de la trajectoire *j*. On note $T^{(i)}_{y_{j,réelle}}$ la valeur pour la coordonnée y du *i-ème* point de la trajectoire *j*. On désigne par $|lot|$ le nombre de trajectoires dans un lot. On désigne par $|T_{j,réelle}|$ la longueur d'une trajectoire.

$$SE(x_1, x_2) = (x_1 - x_2)^2 \tag{3.8}$$

$$L_{trajectoire}(T_{j,réelle}, T_{j,pred}) = \sum_{i=1}^{|T_{j,réelle}|} (SE(T^{(i)}_{x_{j,réelle}}, T^{(i)}_{x_{j,pred}}) + SE(T^{(i)}_{y_{j,réelle}}, T^{(i)}_{y_{j,pred}})) \tag{3.9}$$

$$L_{lot} = \frac{1}{|lot|} \sum_{j=1}^{|lot|} L_{trajectoire}(T_{j,réelle}, T_{j,pred}) \tag{3.10}$$

Tous les modèles sont entraînés avec la même fonction de perte. Pour tous les modèles, on utilise la fonction d'activation *Relu* (équation 2.1.4).

3.5.3 Optimisation des hyper-paramètres

Pour optimiser les hyper-paramètres des modèles, on utilise la méthode de recherche aléatoire de Bergstra et al. [70]. Pour chaque hyper-paramètre, on fixe un domaine de valeur

qu'il peut prendre. On entraîne le modèle plusieurs fois pendant un certain nombre d'itérations en choisissant chaque hyper-paramètre au hasard. On conserve les hyper-paramètres ayant donné le meilleur résultat. Cette méthode permet une bonne exploration de l'espace des hyper-paramètres sans tester toutes les combinaisons d'hyper-paramètres ce qui serait prohibitif pour un grand nombre d'hyper-paramètres. Dans notre étude, pour chaque modèle on réalise deux recherches aléatoires, une première avec des domaines assez larges pour les hyper-paramètres et une deuxième avec des domaines plus restreints étant donnés les résultats de la première recherche. Pour chaque ensemble aléatoire d'hyper-paramètres, on entraîne le modèle pour cinquante itérations sur l'ensemble d'entraînement. Les résultats sont évalués sur l'ensemble de validation. Une fois le meilleur ensemble d'hyper-paramètres trouvés, on ré-entraine le modèle avec ces hyper-paramètres sur l'ensemble d'entraînement et l'ensemble de validation pendant un nombre maximal de cents itérations (valeur choisie arbitrairement). Le modèle final est évalué sur l'ensemble de test.

3.5.4 Arrêt précoce

Pour éviter le sur-apprentissage et réduire le temps d'entraînement, on utilise les méthodes d'arrêt précoce proposées par Prechelt et al. [71]. Ces méthodes permettent d'arrêter l'entraînement quand la fonction de perte du modèle sur l'ensemble d'entraînement cesse d'évoluer ou quand la fonction de perte du modèle sur l'ensemble de validation commence à remonter (signe de sur-apprentissage).

Lors de l'apprentissage, on conserve le modèle issu de l'itération ayant donné la meilleure valeur de la fonction de perte sur l'ensemble de validation. On ne conserve pas nécessairement celui résultant de la dernière itération.

3.6 Les modalités de test final des modèles

Pour rappel, les métriques utilisées dans la majorité des études de la revue de littérature sont le *Average Displacement Error(ADE)* et le *Final Displacement Error(FDE)*. Elles ont déjà été définies dans le cadre de la revue de littérature.

Dans les études proposées dans la revue de littérature, l'évaluation finale des modèles se fait indépendamment trajectoire par trajectoire. C'est-à-dire que même si on prend en compte le contexte d'une trajectoire pour réaliser une prédiction, on évalue la trajectoire prédite indépendamment de son contexte. On ne met pas en relation cette prédiction avec les prédictions des autres trajectoires se déroulant pendant la même période de temps. Cette manière d'évaluer ne permet pas d'évaluer par exemple l'intérêt d'un modèle entraîné selon

le principe d'optimisation conjointe et celui d'un modèle entraîné trajectoire par trajectoire.

Dépendamment de la manière dont le modèle a été entraîné, on mène l'évaluation finale des modèles de manière différente. Pour chaque exemple d'entraînement, on va prédire les trajectoires de l'agent principal et de ses voisins. Pour un modèle à *prédiction non-conjointe*, on effectuera la prédiction de chacune des trajectoires l'une après l'autre. Pour un modèle à *prédiction conjointe*, on effectuera la prédiction de toutes les trajectoires de l'exemple d'entraînement simultanément. On peut ainsi évaluer la capacité du modèle à prédire un ensemble de trajectoires interagissant ensemble de manière cohérente.

De plus, il est possible que les métriques *ADE* et *FDE* ne soient pas suffisantes pour évaluer l'intérêt d'un modèle prenant en compte le contexte de prédiction et un modèle naïf. Étant donné un contexte de prédiction, un agent peut prendre plusieurs décisions possibles. Ainsi, il est peut-être plus intéressant d'évaluer la capacité du modèle à prédire un ensemble de trajectoires interagissant de manière cohérente entre elles et avec la structure physique de la scène que d'évaluer la capacité du modèle à deviner laquelle des prédictions possibles est celle qui va se produire étant donné un contexte de prédiction. En outre, les métriques *ADE* et *FDE* ne permettent pas d'évaluer le réalisme des vitesses et accélérations prédites par les modèles.

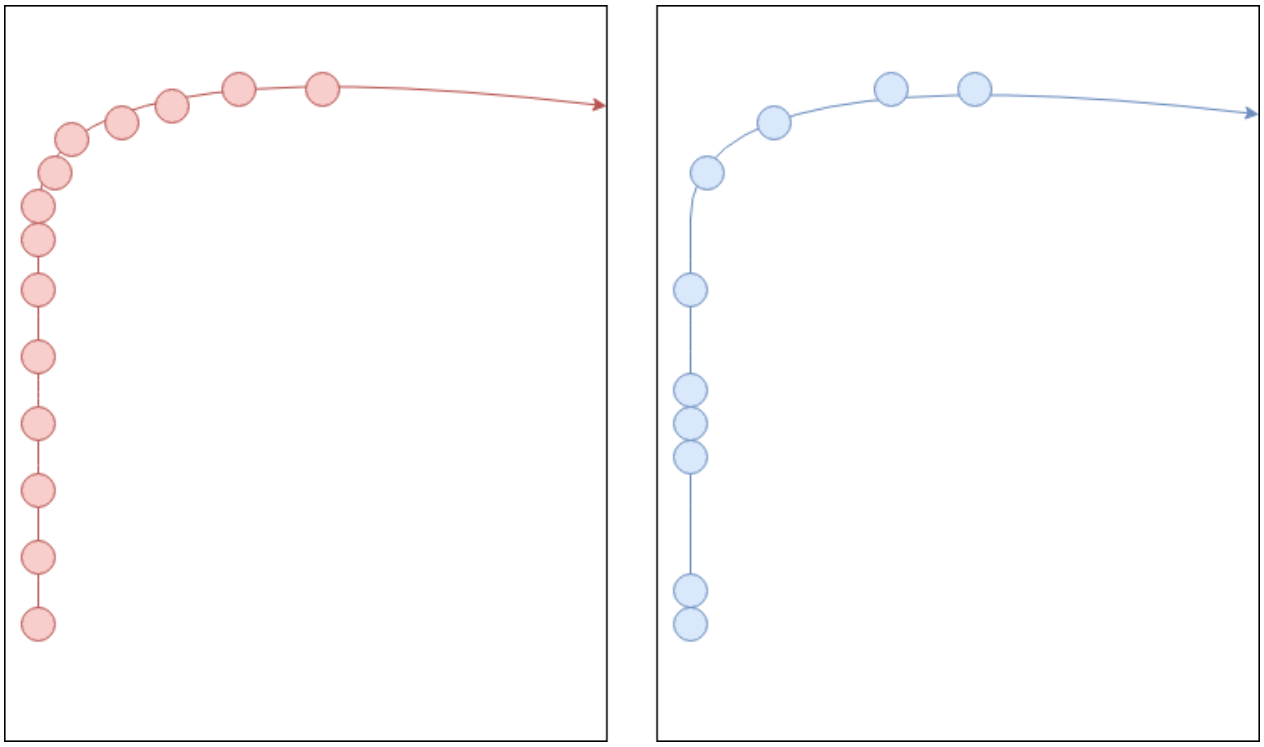


Figure 3.10 Un exemple de différence de cinétique pour deux trajectoires prenant place sur une même courbe : À gauche un écart entre les points plus réaliste qu'à droite, pour une courbe cependant identique.

En effet, admettons que le modèle soit capable de prédire une trajectoire ayant la même forme et la même direction que la trajectoire réelle. Si il place les points de manière ordonnée mais aléatoire le long de cette forme, il aura très certainement un bon résultat (en terme d'*ADE* et *FDE*). Néanmoins, la cinématique de la trajectoire prédite pourrait s'avérer complètement irréaliste (figure 3.10). Un modèle prédisant un autre chemin parmi les différentes possibilités offertes par le contexte, mais avec une cinématique réaliste sera considéré comme moins bon que le modèle précédent alors qu'il a mieux intégré en soi les caractéristiques de déplacement des agents.

Ces constats nous conduisent à proposer des métriques supplémentaires pour évaluer la qualité finale des prédictions effectuées par les différents modèles.

3.6.1 La métrique de réalisme cinétique

Pour juger du réalisme de la dynamique d'une trajectoire, on en étudie deux aspects, les accélérations et les vitesses. Pour l'ensemble des exemples de test, on regroupe les vitesses

réelles et les vitesses prédites afin de représenter ces deux distributions de manière discrète. On mesure la distance entre ces deux distributions à l'aide de la distance de Wasserstein (utilisée notamment par Arjovsky et al. [72]). Intuitivement, cette distance entre deux distributions à une dimension correspond au coût minimum pour transformer une distribution en l'autre. Il s'agit d'une vraie distance, par conséquent elle est symétrique. On utilise la même métrique pour les accélérations.

3.6.2 La métrique pour les interactions sociales

Pour juger de la capacité du modèle à prendre en compte les interactions entre les différents agents, Sadeghian et al. [6] proposent de compter le nombre de fois où deux agents rentrent presque en collision sur la totalité des prédictions de l'ensemble de test. Ils définissent le fait de rentrer presque en collision, quand deux agents sont situés à moins de 0.10 m l'un de l'autre. On reprend la même idée de la manière suivante : pour l'ensemble des trajectoires d'un ensemble d'entraînement (agents principal et voisins), on utilise le modèle pour prédire leurs futures trajectoires. Si le modèle est entraîné de manière conjointe, alors toutes les futures trajectoires sont prédites en une seule fois, sinon on prédit les trajectoires une à une en permutant l'agent principal. La métrique correspond à la moyenne sur toutes les unités de temps du pourcentage de paires d'agents se trouvant dans un cercle dont le diamètre est défini manuellement. Comme la valeur limite de 0.10 m est choisie arbitrairement, on regardera les résultats pour des valeurs de 0.10 m, 0.5 m et 1.0 m.

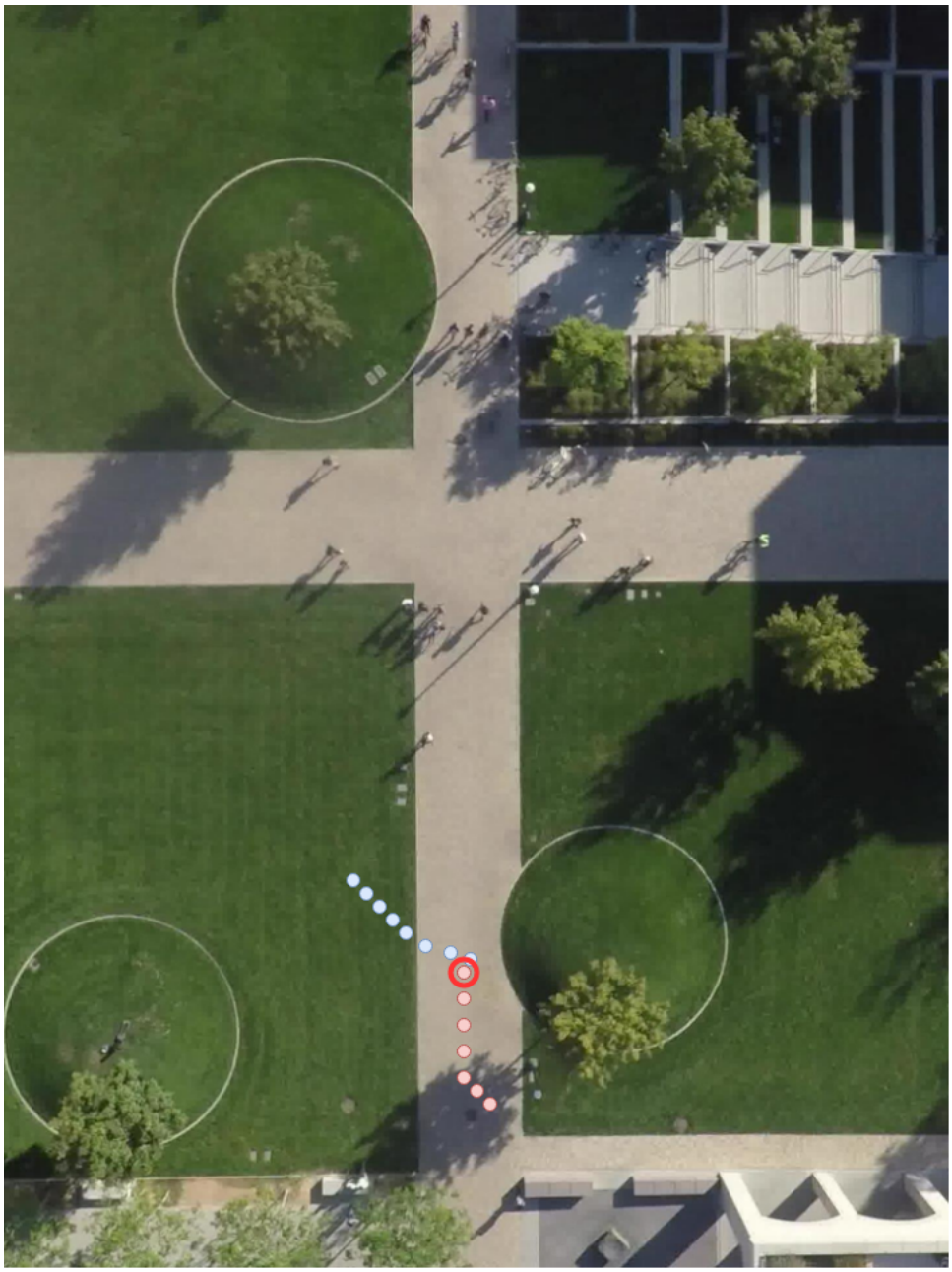


Figure 3.11 Un exemple de conflit entre deux agents

3.6.3 La métrique pour les interactions spatiales

Pour juger de la capacité du modèle à prendre en compte les interactions entre un agent et la structure spatiale de la scène dans laquelle il évolue, on propose de compter le nombre de fois où un agent rentre en collision avec des parties de la scène qu'il ne peut pas traverser ou des obstacles. À cet effet, pour chaque scène de l'ensemble de test, on étiquette manuellement

un masque permettant de sélectionner les parties de l'image ne pouvant pas être traversées. Les contraintes spatiales ne s'appliquent pas de manière identique sur tous les types d'usagers. Par exemple, un piéton a tendance à prendre plus de liberté avec la signalisation qu'une voiture dont le déplacement est très contraint. On définit trois masques par scène, un par groupe d'agents aux caractéristiques de déplacement semblables. On définit trois classes, les obstacles pour les piétons, les obstacles pour les voitures et véhicules répondants aux mêmes contraintes (charriots et bus) et les obstacles pour les vélos et véhicules aux mêmes contraintes (skateboards). On calcule le pourcentage de positions, sur la totalité des positions prédites à partir de l'ensemble de test, entrant en intersection avec un masque. Pour une position prédite, elle est convertie de mètres en pixels puis, dépendamment de son type, le masque associé est choisi et l'on regarde si elle rentre en intersection avec les obstacles manuellement annotés. On reporte la proportion sur l'ensemble des points prédits par le modèle sur l'ensemble de test.

Figure 3.12 Un exemple d'une scène(à gauche) et de son masque pour les voitures(à droite)

Sur la figure 3.12, les zones recouvertes en vert sur l'image de droite correspondent aux zones de la scène qu'une voiture ne peut pas traverser.

Cette métrique dépend à la fois de la définition manuelle des masques et de l'étiquetage des types d'agents fournis dans l'ensemble de données. L'étiquetage manuel peut être source d'erreur dans les résultats de la métrique. Afin de confirmer les conclusions issues de cette

métrique, on propose dans l'annexe B une deuxième métrique spatiale ne nécessitant pas d'étiquetage manuel

3.6.4 Mesure du temps de prédiction

Enfin, on est intéressé à connaître le temps de calcul nécessaire pour la prédiction par chaque modèle d'une seule trajectoire. On reporte le temps moyen mis par trajectoire par chaque modèle sur la totalité de l'ensemble de test.

CHAPITRE 4 RÉSULTATS THÉORIQUES ET EXPÉRIMENTAUX

Dans cette partie, nous effectuons la liste des modèles utilisés en leur associant à chacun un nom. Nous faisons un rappel des différentes métriques utilisées et proposées puis nous réalisons ensuite la liste des expériences et leurs objectifs respectifs. Enfin, nous présentons et analysons les résultats. Les temps de calcul de chaque modèle seront comparés dans une expérience propre, séparée de l'évaluation des performances de prédiction.

Dans tous les tableaux de résultats proposés, on retrouve sur les colonnes les noms des modèles étudiés et en ligne les différentes métriques utilisées. Dans une cellule en gras, on retrouve la meilleur valeur pour une métrique parmi les différents modèles étudiés.

4.1 Modèles

Pour chaque modèle, on utilise en entrée les positions absolues et on prédit les positions relatives à la dernière position observée. En effet, on constate lors de la recherche aléatoire que ces différentes possibilités d'entrées et de sorties n'ont pas d'influence évidente sur les résultats.

- *cnn-mlp* : le modèle de base naïf utilisant un simple *CNN*.
- *rnn-mlp* : le modèle de base naïf utilisant un *LSTM*.
- *s2s-social-j* : le modèle de base attentif utilisant un *LSTM* et un module de *soft-attention* pour les interactions agent/agent.
- *s2s-spatial* : le modèle de base attentif utilisant un *LSTM* et un module de *soft-attention* pour les interactions agent/espace.
- *c-social-soft-j* : le modèle de base attentif utilisant un *CNN* et un module de *soft-attention* pour les interactions agent/agent.
- *c-spatial-soft* : le modèle de base attentif utilisant un *CNN* et un module de *soft-attention* pour les interactions agent/espace.
- *c-social-mha-h-j* : le modèle de base attentif utilisant un *CNN* et un module d'attention multi-tête pour les interactions agent/agent.
- *c-spatial-mha-h* : le modèle de base attentif utilisant un *CNN* et un module d'attention multi-tête pour les interactions agent/espace.

avec pour significations :

- j : indique si le modèle est entraîné de manière conjointe ou non. C'est-à-dire si il apprend à prédire conjointement toutes les trajectoires d'un exemple d'entraînement ou seulement celle de l'agent principal. Le symbole j prend la valeur 0 pour un entraî-

nement disjoint et 1 pour un entraînement conjoint. Cette caractéristique ne concerne que les modèles prenant en compte les interactions sociales. Pour tous les autres, j prend la valeur 0.

— h : indique le nombre de têtes d'attention utilisées pour le modèle. Cette caractéristique ne s'applique que pour les modèles utilisant le mécanisme d'attention multi-tête.

Pour mieux illustrer le système de dénomination, voici quelques exemples :

— *c-social-mha-8-1* : désigne un modèle convolutif attentif utilisant un module d'attention multi-tête pour prendre en compte les interactions agent/agent. Le module utilise huit têtes d'attention et prédit conjointement les futures trajectoires de tous les agents d'un exemple d'entraînement.

— *s2s-social-0* : désigne un modèle de base récurrent attentif prenant en compte les interactions sociales. Le module est entraîné de manière disjointe, il prédit uniquement la trajectoire de l'agent principal en utilisant la trajectoire passée de l'agent principal et celles de ses voisins.

4.2 Métriques

Pour rappel, on a défini les métriques suivantes :

— *ADE* correspond pour la trajectoire prédite de l'agent principal à la moyenne sur toutes les positions, des distances euclidiennes point à point entre la trajectoire réelle et la trajectoire prédite. Elle est exprimée en mètre.

— *FDE* correspond pour la trajectoire prédite de l'agent principal à la distance euclidienne entre le dernier point de la trajectoire réelle et celui de la trajectoire prédite. Elle est exprimée en mètre.

— *ADE (conjointe)* correspond pour la trajectoire prédite de l'agent principal et celles de ses voisins à la moyenne sur toutes les positions, des distances euclidiennes point à point entre la trajectoire réelle et la trajectoire prédite. Elle est exprimée en mètre.

— *FDE(conjointe)* correspond pour la trajectoire prédite de l'agent principal et celles de ses voisins à la distance euclidienne entre le dernier point de la trajectoire réelle et celui de la trajectoire prédite. Elle est exprimée en mètre.

— *Cinétique (vitesses)* correspond à la distance de *Wasserstein* entre la distribution réelle des vitesses observée dans l'ensemble de données et la distribution apprise par le modèle.

— *Cinétique (accélérations)* correspond à la distance de *Wasserstein* entre la distribution réelle des accélérations observée dans l'ensemble de données et la distribution apprise par le modèle.

— La métrique *Sociale (d mètres)* correspond pour l'ensemble des trajectoires d'un ensemble d'entraînement (agents principal et voisins), à la moyenne sur toutes les unités de temps du pourcentage de paires d'agents se trouvant dans un cercle dont le diamètre d est défini manuellement. On reporte les valeurs pour des diamètres de 0.1, 0.5 et 1.0 mètres.
— *Spatiale* correspond pour l'ensemble de toutes les positions prédites par un modèle au pourcentage de ces derniers rentrant en collision avec les zones non traversables de l'environnement.

On nomme métriques de position les métriques *ADE* et *FDE*, métriques cinétiques les métriques *Cinétique (vitesses)* et *Cinétique (accélérations)*, et métriques sociales les métriques *Sociale (0.1 m)*, *Sociale (0.5 m)* et *Sociale (1.0 m)*.

Tous les modèles seront évalués par l'ensemble des métriques à l'exception de *ADE (conjointe)* et *FDE (conjointe)* qui seront utilisées exclusivement pour l'évaluation des modèles entraînés de manière conjointe.

4.3 Expériences

4.3.1 Évaluation des modèles de base naïfs

Description de l'expérience

Dans cette expérience, on compare les deux modèles de base naïfs :
— *cnn-mlp*
— *rnn-mlp*
Cette expérience cherche à vérifier l'équivalence entre les modèles naïfs récurrents et convolutifs naïfs pour la tâche de prédiction de trajectoire. Dans cette expérience, on cherche à évaluer l'hypothèse suivante :

Hypothèse 1 : Utiliser le paradigme convolutif maintient une qualité de prédiction équivalente à l'utilisation du paradigme récurrent.

Résultats

Tableau 4.1 Résultats de l'expérience pour comparer les modèles de base naïfs

	cnn-mlp	*rnn-mlp*
ADE	**1.234**	1.3441
FDE	**2.6679**	2.8051
Cinétique (vitesses)	0.337	**0.3306**
Cinétique (accélérations)	**0.258**	0.2788
Sociale (0.1m)	**1.0922**	1.0937
Sociale (0.5m)	**1.8102**	1.9436
Sociale (1.0m)	4.3794	**4.2857**
Spatiale	25.4773	**25.3981**

Le modèle de base convolutif *cnn-mlp* entraîne une diminution de 8.19% et 4.89% respectivement pour les métriques *ADE*/*FDE*. Pour la métrique *Cinétique (vitesses)*, le modèle convolutif entraîne une augmentation de 1.9%. Pour la métrique *Cinétique (accélérations)*, le modèle convolutif entraîne une diminution de 7.4%. Pour une distance de 0.1 m, les deux modèles ont des performances similaires. Pour une distance de 0.5 m, le modèle convolutif diminue la proportion de conflits entre agents de 6.8%. Pour une distance de 1.0 m, les deux modèles ont des résultats semblables avec une augmentation de 2.2% du *cnn-mlp* par rapport au *rnn-mlp*. Il n'y a pas de différence notable entre les deux modèles pour la métrique *Spatiale*.

Hypothèse 1 : Les résultats présentés montrent que le modèle convolutif et meilleur ou équivalent pour tous les critères sauf pour la distribution des vitesses mais de manière négligeable. On peut donc confirmer qu'utiliser un modèle convolutif pour la tâche de prédiction de trajectoire ne dégrade pas la qualité de la prédiction par rapport à un modèle récurrent et l'améliore même.

4.3.2 Comparaison des modèles de base attentifs récurrents avec le modèle de base naïf récurrent

Description de l'expérience

Dans cette expérience, on compare les deux modèles de base attentifs prenant en compte respectivement les interactions agent/agent et agent/espace avec le modèle de base récurrent :

— *s2s-social-0*
— *s2s-spatial*
— *rnn-mlp*

Cette expérience recouvre deux objectifs. Le premier objectif est de vérifier l'intérêt du mécanisme d'attention proposé par Sadeghian et al. [6] sur la prise en compte des interactions sociales dans le paradigme récurrent. Le deuxième est de vérifier l'intérêt du mécanisme d'attention proposé par Sadeghian et al. [6] sur la prise en compte des interactions spatiales dans le paradigme récurrent. Dans cette expérience, on cherche à évaluer les hypothèses suivantes :

Hypothèse 2 : Utiliser un module d'attention pour prendre en compte les interactions sociales dans le paradigme récurrent permet d'améliorer la qualité de la prédiction.

Hypothèse 3 : Utiliser un module d'attention pour prendre en compte les interactions spatiales dans le paradigme récurrent permet d'améliorer la qualité de la prédiction.

Résultats

Tableau 4.2 Résultats de la comparaison entre modèles de base naïfs et attentifs construits sur le paradigme récurrent

	rnn-mlp	*s2s-social-0*	*s2s-spatial*
ADE	**1.3441**	1.4262	1.4904
FDE	**2.8051**	2.8723	3.0211
Cinétique (vitesses)	**0.3306**	0.3394	0.3685
Cinétique (accélérations)	**0.2788**	0.3029	0.3289
Sociale (0.1m)	1.0937	**1.0613**	1.0834
Sociale (0.5m)	1.9436	**1.6063**	1.7803
Sociale (1.0m)	4.2857	**3.4032**	4.0866
Spatiale	25.3981	25.0261	**24.9969**

Par rapport au modèle de base récurrent *rnn-mlp*, le modèle de base attentif pour les interactions sociales *s2s-social-0* dégrade les performances pour les métriques *ADE* et *FDE* avec une augmentation de 6.10% et 2.39% respectivement. Pour la métrique *Cinétique (vitesses)*, *s2s-social-0* entraîne une augmentation de 2.66%. Pour la métrique *Cinétique (accélérations)*, *s2s-social-0* entraîne une augmentation de 8.64%. Pour une distance de 0.1 m, *s2s-social-0* entraîne une réduction de 2.9% de la proportion des conflits entre agents. Tandis que pour des distances de 0.5 m et 1.0 m, *s2s-social-0* réduit cette proportion de 17.3% et 20.5% respectivement. Il n'y a pas de différence notable entre les deux modèles pour la métrique *Spatiale*.

Hypothèse 2 : Pour toutes les métriques sauf les métriques sociales et spatiales, le modèle d'attention spatial récurrent dégrade les performances marginalement sur certaines (*FDE*,*Cinétique (vitesses)*) et de manière plus prononcée pour d'autres (*ADE*, ,*Cinétique*

(accélérations)). En revanche, les métriques sociales mettent en avant l'impact d'un tel modèle. Si la réduction de la proportion des conflits est minime pour une distance de 0.1 m, elle est très significative pour des distances de 0.5 m et 1.0 m. Il est difficile de conclure quant à une supériorité d'un des deux modèles. Cependant, l'impact du modèle utilisant l'attention social est manifeste.

De son côté le modèle de base attentif pour les interactions spatiales (*s2s-spatial-0*), conduit à une augmentation de 10.88% pour l'*ADE* et de 7.77% pour la *FDE* par rapport à *rnn-mlp*. Pour les métriques *Cinétique (vitesses)* et *Cinétique (accélérations)*, *s2s-spatial-0* entraîne une augmentation de 11.4% et 17.9% respectivement. Pour une distance de 0.1 m, *s2s-spatial-0* ne fait pas de différence avec le modèle de base. Tandis que pour des distances de 0.5 m et 1.0 m, *s2s-spatial-0* réduit cette proportion de 8.4% et 4.6% respectivement. Il n'y a pas de différence notable entre les deux modèles pour la métrique *Spatiale*.

Hypothèse 3 : Le modèle utilisant l'attention dégrade significativement la performance de prédiction pour les métriques *ADE*, *FDE*, *Cinétique (vitesses)* et *Cinétique (accélérations)*. La métrique *Spatiale* ne permet pas de mettre en avant une quelconque prise en compte des interactions spatiales. On peut noter une influence significative du modèle sur les métriques sociales. Ce résultat est surprenant et peut être révélateur de l'utilisation par le modèle des informations spatiales. Globalement, la qualité de prédiction est dégradée et l'influence de l'information spatiale difficilement quantifiable.

4.3.3 Comparaison des modèles de base attentifs convolutifs avec le modèle de base naïf convolutif

Description de l'expérience

Cette expérience est similaire à la précédente. L'étude est la même mais cette fois on transpose l'étude dans le paradigme convolutif. On étudie les modèles suivants :

— *c-social-soft-0*
— *c-spatial-soft*
— *cnn-mlp*

Cette expérience recouvre deux objectifs. Le premier objectif est de vérifier l'intérêt du mécanisme d'attention proposé dans notre étude pour la prise en compte des interactions sociales dans le paradigme convolutif. Le deuxième est de faire la même chose pour le module d'attention spatiale convolutif. Dans cette expérience, on cherche à évaluer les hypothèses suivantes :

Hypothèse 4 : Utiliser un module d'attention pour prendre en compte les interactions sociales dans le paradigme convolutif permet d'améliorer la qualité de la prédiction.

Hypothèse 5 : Utiliser un module d'attention pour prendre en compte les interactions spatiales dans le paradigme convolutif permet d'améliorer la qualité de la prédiction.

Tableau 4.3 Résultats de la comparaison entre modèles de naïfs et attentifs construits sur le paradigme convolutif

	cnn-mlp	*c-social-soft-0*	*c-spatial-soft*
ADE	1.2340	**1.2002**	1.3085
FDE	2.6679	**2.5551**	2.7115
Cinétique (vitesses)	0.3370	**0.2886**	0.3089
Cinétique (accélérations)	0.2580	**0.2331**	0.3012
Sociale (0.1m)	1.0922	**1.0624**	1.0961
Sociale (0.5m)	1.8102	**1.5625**	1.8276
Sociale (1.0m)	4.3794	**3.4442**	4.2762
Spatiale	25.4773	25.3481	**25.3205**

Par rapport au modèle de base récurrent *cnn-mlp*, le modèle attentif proposé pour les interactions sociales *c-social-soft-0* améliore les performances pour les métriques *ADE* et *FDE* en les réduisant de 2.7% et 4.4% respectivement. Pour la métrique *Cinétique (vitesses)*, *c-social-soft-0* entraîne une réduction de 14.4%. Pour la métrique *Cinétique (accélérations)*, *c-social-soft-0* entraîne une diminution de 9.6%. Pour une distance de 0.1 m, *c-social-soft-0* entraîne une réduction de 2.7% de la proportion des conflits entre agents. Tandis que pour des distances de 0.5 m et 1.0 m, *c-social-soft-0* réduit cette proportion de 13.6% et 21.3% respectivement. Il n'y a pas de différence notable entre les deux modèles pour la métrique *Spatiale*.

Hypothèse 4 : Le modèle convolutif avec attention sociale est meilleur que sa contrepartie naïve pour toutes les métriques à l'exception de la métrique *Spatiale*. Marginalement pour *ADE*, *FDE* et *Sociale (0.1 m)* et de manière importante pour les métriques cinétiques et les métriques *Sociale (0.5 m)* et *Sociale (1.0 m)*. L'attention sociale dans le paradigme convolutif améliore très clairement les performances du modèle.

De son côté le modèle de base attentif pour les interactions spatiales *c-spatial-soft*, conduit à une augmentation de 6% pour l'*ADE* et de 1.6% pour la *FDE* par rapport à *cnn-mlp*. Pour les métriques *Cinétique (vitesses)* et *Cinétique (accélérations)*, *c-spatial-soft* entraîne une réduction de 8.3% et une augmentation de 16.7% respectivement. Pour des distances de 0.1 m et 0.5 m, *c-spatial-soft* ne fait pas de différence avec le modèle de base. Tandis que pour une distance de 1.0 m, *c-spatial-soft* réduit cette proportion de 2.3%. Il n'y a pas de différence notable entre les deux modèles pour la métrique*Spatiale*.

Hypothèse 5 : Le modèle convolutif utilisant l'attention spatiale n'a pas d'influence sur

les métriques sociales. Il dégrade de manière significative les métriques *ADE* et *Cinétique (accélérations)* tandis qu'il améliore la métrique *Cinétique (vitesses)*. Globalement, le modèle étudié dégrade légèrement la qualité de prédiction par rapport au modèle de base.

4.3.4 Comparaison pour les modèles attentifs prenant en compte les interactions sociales entre paradigmes récurrents et convolutifs

Description de l'expérience

Dans cette expérience, on compare le modèle de base attentif prenant en compte les interactions sociales avec les deux modèles proposés prenant en compte les mêmes interactions :

- *s2s-social-0*
- *c-social-soft-0*
- *c-social-mha-1-0*

Cette expérience cherche à étudier l'impact de la différence de sémantique induite par le changement de paradigme d'architectures sur la qualité des prédictions. Dans cette expérience, on cherche à évaluer l' hypothèse suivante :

Hypothèse 6 : Le changement de sémantique du module d'attention sociale induit par le passage du paradigme récurrent au paradigme convolutif n'entraîne pas une dégradation de la qualité de prédiction.

Résultats

Tableau 4.4 Résultats de la comparaison entre modèles convolutifs et récurrents utilisant les interactions sociales

	s2s-social-0	*c-social-soft-0*	*c-social-mha-1-0*
ADE	1.4262	**1.2002**	1.206
FDE	2.8723	**2.5551**	2.57
Cinétique (vitesses)	0.3394	**0.2886**	0.297
Cinétique (accélérations)	0.3029	0.2331	**0.232**
Sociale (0.1m)	**1.0613**	1.0624	1.0664
Sociale (0.5m)	1.6063	**1.5625**	1.5786
Sociale (1.0m)	**3.4032**	3.4442	3.4444
Spatiale	**25.0261**	25.3481	25.0937

Dans le tableau 4.4 on compare les performances des deux modèles convolutifs utilisant l'attention sociale *c-social-soft-0* et *c-social-mha-1-0* par rapport à leur contrepartie

récurrente *s2s-social-0*.

Le modèle *c-social-soft-0* diminue de 15.8% et 11.1% l'*ADE* et la *FDE* respectivement. Du côté des métriques cinétiques il diminue de 14.9% la distance entre la distribution des vitesses réelles et celle qu'il a apprise et de 23% pour les accélérations. Il diminue les métriques *Sociale (0.1 m)*, *Sociale (0.5 m)* et *Sociale (1.0 m)* de respectivement 0.1%, 2.7% et 1.2%. On ne note pas d'écart significatif pour la métrique *Spatiale*.

Le modèle *c-social-mha-1-0* donne des résultats très proches de *c-social-soft-0* et ainsi quasiment les mêmes écarts avec les performances de *s2s-social-0*.

Hypothèse 6 : Pour les modèles utilisant l'attention sociale, les modèles convolutifs sont bien meilleurs que leur équivalent récurrent pour les métriques de position (*ADE* et *FDE*) ainsi que les métriques cinétiques. Pour la prise en compte des interactions sociales, mise en évidence par les métriques sociales, les trois modèles sont équivalents. Ainsi, on peut conclure que les deux modèles convolutifs sont meilleurs dans l'absolu que le modèle récurrent.

4.3.5 Comparaison pour les modèles attentifs prenant en compte les interactions spatiales entre paradigmes récurrents et convolutifs

Description de l'expérience

Cette expérience est la même que la précédente mais cette fois prenant en compte les interactions spatiales. Dans cette expérience, on compare le modèle de base attentif prenant en compte les interactions sociales avec les deux modèles proposés prenant en compte les mêmes interactions :

- *s2s-spatial*
- *c-spatial-soft*
- *c-spatial-mha-1*

Cette expérience recouvre deux objectifs. D'abord, étudier l'impact de la différence de sémantique induite par le changement de paradigme d'architectures sur la qualité des prédictions. Ensuite, étudier l'impact de ce changement sur le temps nécessaire à l'entraînement des modèles et à la prédiction d'une trajectoire. Dans cette expérience, on cherche à évaluer l'hypothèse suivante :

Hypothèse 7 : Le changement de sémantique du module d'attention spatiale induit par le passage du paradigme récurrent au paradigme convolutif n'entraîne pas une dégradation de la qualité de prédiction.

Résultats

Tableau 4.5 Résultats de la comparaison entre modèles convolutifs et récurrents utilisant les interactions spatiales

	s2s-spatial	*c-spatial-soft*	*c-spatial-mha-1*
ADE	1.4904	**1.3085**	1.4074
FDE	3.0211	**2.7115**	2.9948
Cinétique (vitesses)	0.3685	**0.3089**	0.3483
Cinétique (accélérations)	0.3289	**0.3012**	0.3689
Sociale (0.1m)	**1.0834**	1.0961	1.0938
Sociale (0.5m)	**1.7803**	1.8276	1.8408
Sociale (1.0m)	**4.0866**	4.2762	4.2932
Spatiale	**24.9969**	25.3205	25.8131

Dans le tableau 4.5 on compare les performances des deux modèles convolutifs utilisant l'attention spatiale *c-spatial-soft* et *c-spatial-mha-1* par rapport à leur contrepartie récurrente *s2s-spatial*.

Le modèle *c-spatial-soft* diminue les métriques *ADE* et *FDE* de 12.2% et 10.2% respectivement par rapport à *s2s-spatial*. De même, il réduit les métriques cinétiques pour les vitesses et les accélérations de 16.2% et 8.4%. Pour les métriques sociales, on constate une augmentation respective de 1.17%, 2.65% et 4.63% pour des distances de 0.1 m, 0.5 m et 1.0 m. Enfin, il n'y a pas d'impact important sur la métrique *Spatiale*, soit une augmentation de 1.29%.

L'autre approche convolutive *c-spatial-mha-1* réduit de 5.56% et 0.8% le couple de métriques *ADE*/*FDE*. Le modèle diminue de 5.5% la distance entre la distribution des vitesses qu'il apprend et la distribution réelle. Tandis qu'il augmente de 12.16% la distance entre les distributions d'accélérations. Il augmente respectivement de 1%, 5% et 3.2% pour les métriques *Sociale (0.1 m)*, *Sociale (0.5 m)* et *Sociale (1.0 m)*. Il entraîne enfin une augmentation de 3.26% pour la métrique *Spatiale*.

Hypothèse 7 : Le modèle convolutif *c-spatial-soft* améliore significativement les performances selon les métriques de position et de cinétique et les dégrade marginalement pour les métriques sociales par rapport au modèle récurrent *s2s-spatial*. Ce modèle est ainsi meilleur que son équivalent récurrent. L'autre approche convolutive *c-spatial-mha-1* améliore les performances selon les métriques de position et les dégrade légèrement selon les métriques sociales. Ce modèle est globalement équivalent à sa contrepartie récurrente. Ainsi, on a proposé un modèle convolutif utilisant l'attention spatiale, ayant de meilleures performances que le modèle récurrent issu de la littérature. Il est important de noter que la métrique *Spatiale*

ne nous permet pas de mettre en évidence la prise en compte des interactions spatiales par aucun des modèles.

4.3.6 Évaluation de l'intérêt d'utiliser plusieurs têtes d'attention pour les interactions sociales

Description de l'expérience

Dans cette expérience, on considère uniquement le modèle d'attention multi-tête *c-social-mha-h-0* pour les interactions sociales. On fait varier le nombre de têtes d'attention afin d'évaluer l'impact de ce dernier sur la qualité de prédiction. Limités par le temps, nous avons comparé uniquement un modèle avec une tête d'attention et un modèle avec quatre têtes d'attention. Les modèles sont entraînés et évalués de manière non-conjointe. Dans cette expérience, on cherche à évaluer l'hypothèse suivante :

Hypothèse 8 : L'utilisation de plusieurs têtes d'attention dans le module d'attention sociale permet une amélioration de la qualité de la prédiction.

Résultats

Tableau 4.6 Impact de l'utilisation de plusieurs têtes d'attention pour les interactions sociales

	c-social-mha-1-0	*c-social-mha-4-0*
ADE	**1.206**	1.2193
FDE	**2.57**	2.6057
Cinétique (vitesses)	0.297	**0.2841**
Cinétique (accélérations)	**0.232**	0.2329
Sociale (0.1m)	1.0664	**1.0637**
Sociale (0.5m)	1.5786	**1.5565**
Sociale (1.0m)	3.4444	**3.4222**
Spatiale	**25.0937**	25.1936

Dans le tableau 4.6 on utilise le modèle d'attention sociale utilisant le produit scalaire comme fonction de score en variant le nombre de têtes d'attention utilisées. On compare un modèle avec quatre têtes d'attention *c-social-mha-4-0* avec *c-social-mha-1-0* un modèle n'en ayant qu'une seule. La seule différence notable est pour la métrique *Cinétique(vitesses)*, le fait d'utiliser quatre têtes d'attention réduit la valeur de la métrique de 4.34%. Pour toutes les autres métriques, *c-social-mha-4-0* augmente d'environ 1% ou moins les différences.

Hypothèse 8 : Les performances des modèles convolutifs utilisant l'attention sociale ne

varient pas significativement quand on fait varier le nombre de têtes d'attention. Le mécanisme d'attention multi-tête ni n'améliore, ni ne dégrade la qualité de la prédiction. Il est possible que les interactions entre agents ne soient pas suffisamment complexes pour qu'il y ait besoin de les considérer selon plusieurs aspects différents (plusieurs têtes d'attention), ce qui expliquerait l'absence d'amélioration.

4.3.7 Évaluation de l'intérêt d'utiliser plusieurs têtes d'attention pour les interactions spatiales

Description de l'expérience

Dans cette expérience, on considère uniquement le modèle d'attention multi-tête *c-spatial-mha-h-0* pour les interactions spatiales. On fait varier le nombre de têtes d'attention afin d'évaluer l'impact de ce dernier sur la qualité de prédiction. Dans cette expérience, on cherche à évaluer l'hypothèse suivante :

Hypothèse 9 : L'utilisation de plusieurs têtes d'attention dans le module d'attention spatiale permet une amélioration de la qualité de la prédiction.

Résultats

Tableau 4.7 Impact de l'utilisation de plusieurs têtes d'attention pour les interactions spatiales

	c-spatial-mha-1	*c-spatial-mha-4*
ADE	1.4074	**1.3627**
FDE	2.9948	**2.9088**
Cinétique (vitesses)	0.3483	**0.3362**
Cinétique (accélérations)	0.3689	**0.3388**
Sociale (0.1m)	**1.0938**	1.0941
Sociale (0.5m)	1.8408	**1.8307**
Sociale (1.0m)	**4.2932**	4.3281
Spatiale	25.8131	**25.362**

Dans le tableau 4.7 on étudie l'impact sur les modules d'attention spatiale d'utiliser une tête d'attention (*c-spatial-mha-1*) ou quatre têtes d'attention (*c-spatial-mha-4*).

Utiliser quatre têtes d'attention réduit l'*ADE* de 3.2% et la *FDE* de 2.9%. Cela réduit aussi les métriques jugeant de la qualité cinétique de 3.47% et 8.2% pour les vitesses et les accélérations respectivement. Pour toutes les autres métriques, la différence entre les deux modèles est inférieure ou proche de 1%.

Hypothèse 9 : L'utilisation de plusieurs têtes d'attention dans le module d'attention spatiale permet une amélioration marginale de la qualité de la prédiction. On n'observe aucune influence mesurée par la métrique *Spatiale*.

4.3.8 Évaluation de l'intérêt d'utiliser une prédiction conjointe pour les interactions sociales

Description de l'expérience

Dans cette expérience on prend en compte un sous-ensemble des modèles. On se concentre sur le paradigme convolutif. On considère le modèle de base convolutif (*cnn-mlp*). Ainsi qu'un modèle convolutif utilisant les interactions sociales parmi ceux proposés, entraîné de manière normale(*c-social-soft-0*) et de manière conjointe(*c-social-soft-1*). Pour rappel, la prédiction conjointe consiste à apprendre au modèle à prédire simultanément les futures trajectoires de l'agent principal et de ses voisins. Ce procédé est pertinent uniquement pour les modèles prenant en compte les interactions entre agents. Dans cette expérience, on veut évaluer l'impact sur la tâche de prédiction de réaliser conjointement les prédictions. Dans cette expérience, on cherche à évaluer l'hypothèse suivante :

Hypothèse 10 : Apprendre et prédire conjointement les futures trajectoires d'un exemple d'entraînement permet de mieux prendre en compte le contexte social lors de la prédiction.

Résultats

Tableau 4.8 Impact de l'optimisation conjointe pour les modèles convolutifs

	cnn-mlp	*c-social-soft-0*	*c-social-soft-1*
ADE	1.2340	1.2002	**1.1791**
ADE(conjointe)	1.2200	2.8674	**1.1643**
FDE	2.6679	2.5551	**2.5442**
FDE(conjointe)	2.5649	5.1346	**2.438**
Cinétique (vitesses)	0.3370	0.2886	**0.2809**
Cinétique (accélérations)	0.2580	**0.2331**	0.2536
Sociale (0.1m)	1.0922	**1.0624**	1.0939
Sociale (0.5m)	1.8102	**1.5625**	1.8593
Sociale (1.0m)	4.3794	**3.4442**	4.437
Spatiale	25.4773	25.3481	**25.2866**

Dans le tableau 4.8, on compare les performances du modèle de base convolutif *cnn-mlp* avec le modèle convolutif d'attention sociale entraîné de manière disjointe *c-soft-attention-0* et de manière conjointe *c-soft-attention-1*.

En terme d'*ADE* et *FDE*, l'entraînement conjoint donne des performances équivalentes à l'entraînement non conjoint soit une réduction d'environ 4.5% par rapport au modèle de base. Pour *ADE (conjointe)* et *FDE (conjointe)*, le modèle entraîné de manière disjointe obtient des performances bien moins bonnes que le modèle de base en multipliant l'erreur par plus de 2 pour chacune des deux métriques. Le modèle entraîné de manière conjointe réduit le couple *ADE (conjointe)/FDE (conjointe)* de 4.5% et 5% respectivement. Pour la métrique des relations sociales par contre, *c-soft-attention-1* ne fait quasiment aucune différence par rapport au modèle de base. Enfin, on n'observe pas de différence notable pour la métrique *Spatiale*.

Hypothèse 10 : La prédiction conjointe améliore légèrement les performance selon les métriques de position calculées de manière disjointe et améliore grandement les performances selon les métriques de positions calculées de manière conjointe. Ainsi, pour les métriques de position, entraîner le modèle à prédire de manière conjointe toutes les trajectoires d'un exemple d'entraînement est bénéfique. En revanche, l'optimisation conjointe annule complètement les effets du module d'attention sur les métriques sociales. L'apprentissage conjoint ne permet donc pas une amélioration de la prise en compte du contexte social.

4.3.9 Évaluation du temps nécessaire à la prédiction d'une trajectoire pour chaque modèle

Description de l'expérience

Dans cette expérience, on répertorie pour chaque modèle le temps nécessaire à la prédiction d'une seule trajectoire. Les valeurs de temps reportées correspondent au temps moyen de prédiction d'une trajectoire pour chaque modèle, sur l'ensemble des exemples de l'ensemble de test. On cherche à vérifier plusieurs hypothèses :

Hypothèse 11 : Le modèle de base convolutif *cnn-mlp* est plus rapide que le modèle de base récurrent *rnn-mlp*.

Hypothèse 12 : Les modèles utilisant l'attention sociale basée sur le paradigme convolutif sont plus rapides que ceux basés sur le paradigme récurrent.

Hypothèse 13 : Les modèles utilisant l'attention spatiale basée sur le paradigme convolutif sont plus rapides que ceux basés sur le paradigme récurrent.

Hypothèse 14 : Les modèles utilisant l'attention sociale entraînés de manière conjointe sont plus rapides que ceux entraînés de manière disjointe.

Résultats

Tableau 4.9 Temps de prédiction d'une trajectoire par modèle en millisecondes

Modèles	Temps unitaire (ms)
cnn-mlp	0.14
rnn-mlp	0.17
s2s-social-0	3.39
s2s-social-1	3.00
c-social-soft-0	0.47
c-social-soft-1	0.30
c-social-mha-1-0	0.48
c-social-mha-1-1	0.28
c-social-mha-4-0	0.73
c-social-mha-4-1	0.49
s2s-spatial	2.87
c-spatial-soft	0.31
c-spatial-mha-1	0.28
c-spatial-mha-4	0.44

Dans le tableau 4.9 répertoriant la vitesse de chaque modèle pour prédire une trajectoire, on remarque les choses suivantes :

— Le *cnn-mlp* réduit le temps de prédiction de 17.6% par rapport au *rnn-mlp*.
— Le *c-social-soft-0* divise le temps de calcul par plus de 7 fois par rapport à *s2s-social-0*.
— Le *c-spatial-soft* divise le temps de calcul par plus de 9 fois par rapport à *s2s-spatial*.
— Le *c-social-soft-1* diminue le temps de calcul de 36% par rapport à *c-social-soft-0*.
— Le *s2s-social-1* diminue le temps de calcul de 11.5% par rapport à *s2s-social-0*.

Hypothèse 11 : Le modèle de base convolutif *cnn-mlp* est effectivement significativement plus rapide que le modèle de base récurrent *rnn-mlp*.

Hypothèse 12 : Les modèles utilisant l'attention sociale basée sur le paradigme convolutif sont sept fois plus rapides que ceux basés sur le paradigme récurrent, confirmant ainsi l'hypothèse.

Hypothèse 13 : Les modèles utilisant l'attention spatiale basée sur le paradigme convolutif sont neuf fois plus rapides que ceux basés sur le paradigme récurrent, confirmant ainsi l'hypothèse.

Hypothèse 14 : Les modèles utilisant l'attention sociale entraînés de manière conjointe sont plus rapides que ceux entraînés de manière disjointe au prix de la perte de leur impact sur les interactions sociales.

4.3.10 Bilan

Dans cette partie, on résume les principales contributions apportées par les expériences :

- On a vérifié que pour les modèles de base, le modèle convolutif est meilleur selon toutes les métriques que le modèle récurrent. Il est en outre plus rapide.
- Le modèle d'attention sociale issu du paradigme récurrent dégrade un peu la performance par rapport au modèle de base récurrent, mais dispose d'un impact visible, quantifié par la métrique sociale, sur la réduction du nombre de conflits entre agents.
- Le modèle d'attention spatiale issu du paradigme récurrent dégrade significativement la qualité par rapport au modèle de base récurrent, mais dispose d'un fort impact sur les interactions sociales.
- L'attention sociale dans le paradigme convolutif améliore très clairement les performances du modèle en tous points, et dénote un impact social visible par rapport au modèle de base convolutif. Il est en outre équivalent pour l'impact sur les interactions sociales avec son équivalent récurrent et meilleur sur toutes les autre métriques. Les distributions cinétiques apprises sont elles aussi meilleures pour le modèle convolutif. Il est enfin sept fois plus rapide. L'approche convolutive pour l'attention spatiale obtient ainsi de bien meilleures performances que l'approche récurrente.
- Le modèle convolutif utilisant l'attention spatiale étudié dégrade légèrement la qualité de prédiction par rapport au modèle de base sans montrer de signe d'utilisation des interactions spatiales. Il est en revanche globalement meilleur que le modèle récurrent issu de la littérature. Il est enfin neuf fois plus rapide.
- L'utilisation de plusieurs têtes d'attention n'a pas d'impact significatif ni pour les modèles d'attention sociale ni pour ni pour ceux d'attention spatiale.
- Dans les modèles d'attention sociale issus du paradigme convolutif, l'entraînement conjoint permet de meilleurs résultats pour les métriques de positions jointes et disjointes que le modèle convolutif de base et ou que du même modèle entraîné de manière non-conjointe. L'influence sur les métriques sociales observée pour le modèle entraîné de manière disjointe ne sont en revanche pas retrouvées lorsqu'il est entraîné de manière conjointe. Le modèle entraîné de manière conjointe est plus rapide que celui entraîné de manière non conjointe.
- La métrique pour les interactions spatiales ne nous permet pas de mettre en évidence la prise en compte des interactions spatiales par aucun des modèles. Le taux très

important de points rentrant en conflit avec les obstacles dans les scènes laisse présager un problème au niveau de l'étiquetage des types d'usagers ou de la définition des masques appliqués sur les scènes.

— Les métriques pour les interactions sociales nous permettent de mettre en évidence la prise en compte des interactions sociales par les modèles d'attention sociale.

— Les métriques pour la qualité cinétique des trajectoires prédites par les modèles permet de mettre en évidence des différences de qualité entre les distributions de vitesses et d'accélérations apprises par les modèles.

CHAPITRE 5 CONCLUSION

L'apparition de caméras peu coûteuses pouvant fournir des données relatives aux mouvements des usagers et à la circulation routière ainsi que le développement rapide de l'apprentissage profond ont conduit la recherche à utiliser ces nouvelles données et algorithmes afin de développer des algorithmes permettant de prédire les futures positions d'un agent se déplaçant dans le trafic pouvant être utilisés pour les voitures autonomes ou la sécurité routière. Au fur-et-à-mesure des études, s'est développé progressivement un consensus sur les modalités d'évaluations. De nombreuses méthodes d'apprentissage profond ont été proposées essayant de prendre en compte les interactions d'un agent avec son environnement tant social que spatial.

Parmi ces méthodes, les mécanismes d'attention importés du domaine du traitement du langage naturel ont été transposés par un petit nombre d'études afin d'être utilisés dans le contexte de la prédiction de trajectoires. Ces mécanismes permettent d'entraîner un modèle à sélectionner automatiquement quelles parties de l'information en entrée du modèle est pertinente pour la tâche de prédiction. Utilisés pour prendre en compte les interactions entre agents, ces mécanismes pourraient par exemple sélectionner quels agents voisins ont le plus d'influence sur la future trajectoire d'un agent. Utilisés pour prendre en compte les interactions entre un agent et son environnement spatial, ils pourraient sélectionner quels parties de la scènes ont le plus d'influence sur la future trajectoire. Les mécanismes d'attentions ont été transposés tels quels du domaine du traitement du langage naturel et se basent tous sur le paradigme récurrent issu du traitement des séquences. Dans ce travail, on s'est intéressé à évaluer ces mécanismes d'attention.

On a aussi proposé des mécanismes d'attention plus adaptés spécifiquement au problème construits autour des réseaux de neurones convolutifs et affichant un meilleur temps d'exécution. L'évaluation des ces modèles prenant en compte les interactions d'un agent avec son environnement, nécessite un cadre d'évaluation adapté, que ne nous apporte pas le consensus sur les modalités d'évaluation. Face au constat, d'abord que des modèles ne prenant pas en compte les interactions montrent en général de meilleurs résultats que les modèles les prenant en compte, ensuite que la capacité d'un modèle de prédiction à utiliser et comprendre son environnement ne peut pas être réduite à sa capacité à prédire l'exacte future trajectoire d'un agent, nous avons cherché à redéfinir en partie les modalités d'évaluation. Dans ce cadre là, nous proposons d'utiliser un ensemble de données plus riche en interactions et modalités de déplacement, permettant ainsi de mieux différencier les performances d'un modèle naïf d'un

modèle plus complexe ainsi que trois nouvelles métriques permettant d'évaluer plus en détail la capacité du modèle à prendre des décisions réalistes.

5.1 Synthèse des travaux

Pour résoudre les problèmes évoqués, on a d'abord proposé un nouveau cadre d'évaluation. On utilise le *SDD* de Sadeghian et al. [13] dans sa totalité, c'est-à-dire faisant interagir six types d'agents différents, pour un total de 20k trajectoires sur huit scènes aux structures spatiales variées. Cet ensemble de données dépasse tous les autres faisant consensus en terme de variété interactionnelle. Il avait déjà été utilisé dans un faible nombre d'études mais en prenant en compte un seul type d'usager (les piétons) réduisant drastiquement l'ampleur de sa richesse interactionnelle.

La moyenne des erreurs de déplacement et l'erreur finale de déplacement, les deux métriques faisant consensus évaluent la capacité d'un modèle à prédire exactement la trajectoire observée position par position. Il nous est apparu que c'était insuffisant pour évaluer la capacité d'un modèle à utiliser l'environnement d'un agent pour prédire sa future trajectoire, puisque étant donné un contexte de prédiction (scène, agents voisins...) un agent peut effectuer une large variété de trajectoire en concordance avec les contraintes imposées par cet environnement. Les métriques proposées cherchent à évaluer la capacité d'un modèle à prendre en compte ces contraintes lors d'une prédiction :

— La métrique de dynamisme cinétique évalue le réalisme d'une trajectoire par rapport aux accélérations et vitesses qui la composent.
— La métrique d'interactions sociales (reprise de la littérature) compte la proportion de conflits entre les trajectoires prédites de différents agents à un instant donné. Un conflit correspondant à deux agents dont la distance est inférieure à un seuil.
— La métrique d'interactions spatiales, contenant la proportion de points d'une trajectoire prédite rentrant en conflit avec l'environnement spatial. Un conflit étant l'intersection de la position de l'agent avec une zone de la scène qu'il ne peut normalement pas traverser. Les zones ont été étiquetées manuellement en prenant en compte le type de l'usager.

Dans un deuxième temps, nous avons repris de la littérature deux modèles n'utilisant ni le contexte social ni le contexte spatial. Un modèle dont l'architecture est basée sur les réseaux de neurones récurrents et un dont l'architecture est basée sur les réseaux convolutifs pour comparer leurs performances. On a pu vérifier que le *CNN* permettait une amélioration sur l'ensemble des critères d'évaluation par rapport au *RNN*. Nous avons ensuite voulu évaluer les

modèles d'attention sociale et spatiale basés sur des architectures récurrentes afin de vérifier qu'ils aient un impact sur les métriques nouvellement proposées. La métrique sociale met en évidence l'intérêt du modèle d'attention sociale récurrent. Celui-ci permet une réduction importante de la proportion de conflits entre agents pour des distances de 0.5 m et 1 m démontrant la capacité du modèle à faire usage du contexte social, au prix d'une légère réduction de sa capacité à restituer la trajectoire exacte attendue. La métrique spatiale ne permet pas de mettre en évidence un impact du module d'attention spatiale récurrent sur la prise en compte du contexte spatial, entraînant même de surcroit une dégradation des performances de prédiction sur l'ensemble des critères d'évaluation.

Nous avons ensuite introduit un modèle attentif de prédiction basé sur le paradigme des réseaux de neurones convolutifs. Le changement de type d'architecture, impose une contrainte structurelle sur le module d'attention impactant la sémantique portée par ce module. Plutôt que de recalculer le contexte d'attention pour prédire chaque future position, nous le calculons une unique fois pour l'ensemble de la trajectoire prédite. Par la même, le contexte de décision du module d'attention change puisque la décision d'accorder son attention à certains éléments du contexte ne se base non plus sur les dernières positions prédites mais sur la trajectoire observée. On a décliné ce modèle pour l'attention sociale et l'attention spatiale et évalué l'impact de ce changement sémantique sur ces deux déclinaisons. D'autres variantes de ce modèle ont été évaluées, mais rejetées face à l'absence d'améliorations notables. Comme pour son équivalent récurrent, la métrique spatiale ne permet pas de mettre en évidence une aptitude du module d'attention spatiale proposé à réduire la tendance des trajectoires prédites à parcourir des zones non traversables dans les scènes. Pour le modèle d'attention sociale, on observe un impact important de la prise en compte du contexte social sur la réduction des conflits entre agents. L'impact observé est équivalent à celui observé pour le modèle d'attention sociale récurrent. Par ailleurs, le modèle convolutif est meilleur en tout point que son équivalent récurrent et que le modèle de base convolutif, c'est-à-dire pour les erreurs de déplacement des positions prédites ainsi que la qualité des distributions de vitesses et d'accélérations apprises. Les deux approches, convolutive et récurrente prennent donc en compte les interactions sociales, de manière similaire. Cela confirme en partie l'idée que l'information apportée par le module d'attention dans le paradigme récurrent est redondante. L'intérêt de l'approche convolutive est d'autant plus fort qu'elle permet une division du temps de prédiction par sept. On a donc proposé un modèle prenant en compte les interactions sociales plus complet et plus rapide que le modèle déjà existant. La comparaison des modèles attentifs a été permise par les métriques proposées sans lesquels, il aurait été difficile d'évaluer les performances relatives des modèles sur différents aspects, si elles sont perfectibles, elles constituent un premier pas vers une évaluation plus complète des modèles de prédiction de

trajectoire.

Enfin, on a étudié l'intérêt d'entraîner le modèle convolutif prenant en compte les interactions sociales à prédire les trajectoires de tous les agents présents à un instant donné simultanément. Cette méthode d'entraînement montre une meilleure capacité à réduire les erreurs de déplacement qu'un modèle entraîné à prédire uniquement la trajectoire de l'agent principal. Cette manière d'optimiser une réduction du temps de prédiction de 36% pour les approches convolutives, mais annule l'impact de l'attention sociale sur la prise en compte des conflits sociaux.

5.2 Limitations des solutions proposées

D'abord, les modèles convolutifs proposés fonctionnent mieux dans un cadre où la longueur de la trajectoire en entrée et la longueur de la trajectoire en sortie sont fixes. Dans le cas contraire, il faut fixer des bornes supérieures pour ces longueurs et utiliser du rembourrage pour que toutes les trajectoires aient les mêmes dimensions.

Ensuite, nous n'avons pas pu mettre en évidence un quelconque impact des modèles d'attention spatiale sur l'utilisation de l'information spatiale. Il peut y avoir plusieurs raisons à cela. D'abord, une erreur d'implémentation peut être possible. Ensuite, on peut remettre en cause la métrique spatiale. Si la métrique en elle même est simple et difficilement critiquable quant à sa pertinence, d'autres facteurs peuvent l'affecter comme l'étiquetage manuel des masques. Il est possible qu'ils soient partiellement incorrects, dissimulant ainsi un éventuel impact des modules d'attention spatiale. Il peut aussi y avoir un taux important d'erreurs d'étiquetage des types d'usagers rendant confus le choix du masque à utiliser pour un agent donné (comment est gérée la transition de type d'usager pour un cycliste descendant de son vélo pour continuer sa route en marchant ?). Néanmoins, la prise en compte de l'information spatiale n'a été mise en avant pour aucun autre critère d'évaluation. Pourtant, c'est probablement l'information qui conditionne le plus le déplacement d'un agent au sein d'une scène. Une future approche devrait se concentrer majoritairement sur cet aspect afin d'améliorer de manière importante les performances de prédiction.

L'ensemble de données proposent plusieurs types d'agents différents. On a ignoré cette information lors de l'apprentissage. Pourtant, chaque type d'agent dispose de modalités de déplacement différentes. On pourrait conditionner les modèles sur le type des agents lors de l'entraînement pour utiliser cette information.

L'ensemble de données, bien que plus riche que ceux utilisés dans la majorité des études ne fournit peut-être pas assez d'interactions sociales ayant un impact sur les trajectoires des

usagers. Le fait que les modèles naïfs ne prédisent que 1% à 4% de conflits par unité de temps illustre le manque de ce type d'interactions. Les améliorations apportées par les modèles proposés sont importantes de manière relative, mais dans l'absolu, la différence entre 1.8% et 1.5% de collisions prédites n'est pas très importante. De plus, les scènes de l'ensemble des données proviennent d'un campus universitaire. Les principaux agents présents sont les piétons et les cyclistes. Ces agents dans un tel environnement, ont une grande liberté de mouvement, bien moins contrainte que dans un environnement urbain où les règles sont plus strictes et les modalités de déplacement bien plus limitées. Cette distribution de données est peut-être trop éloignée du champs des applications qui nous intéressent. De même, les variations entre les performances des modèles pour prédire exactement la trajectoire attendue sont dans une fourchette de zéro à environ trente centimètres d'erreur moyenne de déplacement. Aucun des modèles ne permet d'amélioration majeure dans l'absolu et ils plafonnent tous autour des mêmes valeurs.

Plus généralement, la limite principale provient de la manière de poser le problème de prédiction comme un problème de régression dans lequel on limite l'horizon de prédiction à un certain nombre d'unités de temps. D'abord, la distance parcourue pendant un certain nombre d'unités de temps n'est pas la même pour un cycliste ou un piéton. Ainsi, dans un même horizon de prédiction, certains usagers sont amenés à prendre une décision de direction importante tandis que d'autres ne parcourent pas suffisamment de distance pour que soit le cas. Aussi, sur des horizons de prédiction fixe de l'ordre de quelques secondes, la majorité des agents se déplacent tout droit. Dans un tel cas, la majorité des prédictions pourraient être effectuées par un modèle à vitesse ou accélération constante. Ainsi, le découpage en sous-trajectoires n'est pas forcément pertinent. Enfin, le problème de prédiction est un problème probabiliste, qui se rapproche plus de la simulation que de la régression. Ainsi, la validation ne devrait se faire qu'en agrégé, pas trajectoire par trajectoire.

5.3 Améliorations futures

Parmi les améliorations futures, on pourrait :

1. Se concentrer sur des approches plus visuelles, se concentrant sur la valorisation de l'information spatiale. On pourrait par exemple représenter la scène comme une matrice en vue de dessus et les trajectoires dans une simple matrice de zéros et de uns là où les agents sont présents. Et essayer de prédire les futures positions comme une image. Cela permettrait de mettre en correspondance directe les agents avec leur environnement spatial.
2. Essayer pour une scène d'obtenir (manuellement ou de manière automatique) des

cartes de traversabilité de la scène par type d'agent et de les intégrer à l'approche visuelle.

3. Intégrer l'information des types d'usagers dans les modèles appris pour distinguer les différentes modalités de déplacement.
4. Utiliser un ensemble de données issu d'un environnement urbain où les règles de circulation contraignent plus le déplacement des agents.
5. Se concentrer sur la développement de modèles génératifs et le développement de fonctions de pertes probabilistes.

Il est en tout cas primordial de redéfinir la tâche de prédiction de trajectoire ainsi que ses modalités d'évaluation de façon cohérente.

RÉFÉRENCES

[1] F. Deloche. (2017) Unfolded basic recurrent neural network. [En ligne]. Disponible : https://en.wikipedia.org/wiki/Recurrent_neural_network#/media/File:Recurrent_neural_network_unfold.svg

[2] Deloche. (2017) Schéma d'un réseau lstm à une unité. le graphe des opérations est détaillé pour l'étape . les poids ne sont pas indiqués. [En ligne]. Disponible : https://fr.wikipedia.org/wiki/R%C3%A9seau_de_neurones_r%C3%A9currents#/media/Fichier:Long_Short-Term_Memory.svg

[3] S. Bai, J. Z. Kolter et V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv :1803.01271*, 2018.

[4] A. Vaswani *et al.*, "Attention is all you need," dans *Advances in neural information processing systems*, 2017, p. 5998–6008.

[5] N. Nikhil et B. Tran Morris, "Convolutional neural network for trajectory prediction," dans *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, p. 0–0.

[6] A. Sadeghian *et al.*, "Sophie : An attentive gan for predicting paths compliant to social and physical constraints," *arXiv preprint arXiv :1806.01482*, 2018.

[7] Q. Tran et J. Firl, "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression," dans *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, p. 918–923.

[8] B. T. Morris et M. M. Trivedi, "Trajectory learning for activity understanding : Unsupervised, multilevel, and long-term adaptive approach," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, n°. 11, p. 2287–2301, 2011.

[9] L. Ballan *et al.*, "Knowledge transfer for scene-specific motion prediction," dans *European Conference on Computer Vision*. Springer, 2016, p. 697–713.

[10] Y. LeCun *et al.*, "Object recognition with gradient-based learning," dans *Shape, contour and grouping in computer vision*. Springer, 1999, p. 319–345.

[11] N. Lee *et al.*, "Desire : Distant future prediction in dynamic scenes with interacting agents," dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, p. 336–345.

[12] R. Hug *et al.*, "On the reliability of lstm-mdl models for pedestrian trajectory prediction," dans *Proceedings of the VIIth International Workshop on Representation, analysis*

and recognition of shape and motion FroM Image Data (RFMI 2017), Póvoa de Varzim, Portugal, 2017, p. 21–23.

[13] A. Sadeghian *et al.*, "Trajnet : Towards a benchmark for human trajectory prediction," *arXiv preprint*, 2018.

[14] G. E. Hinton et R. S. Zemel, "Autoencoders, minimum description length and helmholtz free energy," dans *Advances in neural information processing systems*, 1994, p. 3–10.

[15] I. Sutskever, O. Vinyals et Q. V. Le, "Sequence to sequence learning with neural networks," dans *Advances in neural information processing systems*, 2014, p. 3104–3112.

[16] S. Hochreiter et J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, nº. 8, p. 1735–1780, 1997.

[17] Y. LeCun *et al.*, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, nº. 11, p. 2278–2324, 1998.

[18] O. Russakovsky *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, nº. 3, p. 211–252, 2015.

[19] K. Simonyan et A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv :1409.1556*, 2014.

[20] J. Long, E. Shelhamer et T. Darrell, "Fully convolutional networks for semantic segmentation," dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, p. 3431–3440.

[21] J. Gehring *et al.*, "Convolutional sequence to sequence learning," dans *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, p. 1243–1252.

[22] D. Bahdanau, K. Cho et Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv :1409.0473*, 2014.

[23] K. Xu *et al.*, "Show, attend and tell : Neural image caption generation with visual attention," *arXiv preprint arXiv :1502.03044*, 2015.

[24] I. Goodfellow *et al.*, "Generative adversarial nets," dans *Advances in neural information processing systems*, 2014, p. 2672–2680.

[25] S. Lefèvre, D. Vasquez et C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH journal*, vol. 1, nº. 1, p. 1, 2014.

[26] S. Ammoun et F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," dans *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*. IEEE, 2009, p. 417–422.

[27] G. Welch, G. Bishop *et al.*, "An introduction to the kalman filter," 1995.

[28] C.-F. Lin et A. Ulsoy, "Vehicle dynamics and external disturbance estimation for future vehicle path prediction," dans *Proceedings of 1995 American Control Conference-ACC'95*, vol. 1. IEEE, 1995, p. 155–159.

[29] A. Eidehall et L. Petersson, "Statistical threat assessment for general road scenes using monte carlo sampling," *IEEE Transactions on intelligent transportation systems*, vol. 9, n°. 1, p. 137–147, 2008.

[30] X. Li, W. Hu et W. Hu, "A coarse-to-fine strategy for vehicle motion trajectory clustering," dans *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 1. IEEE, 2006, p. 591–594.

[31] B. Morris et M. Trivedi, "Learning trajectory patterns by clustering : Experimental studies and comparative evaluation," dans *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, p. 312–319.

[32] Z. Zhang *et al.*, "Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes." dans *ICPR (3)*. Citeseer, 2006, p. 1135–1138.

[33] W. Hu *et al.*, "A system for learning statistical motion patterns," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, n°. 9, p. 1450–1464, 2006.

[34] F. Large *et al.*, "Avoiding cars and pedestrians using velocity obstacles and motion prediction," dans *IEEE Intelligent Vehicles Symposium, 2004*. IEEE, 2004, p. 375–379.

[35] K. Kim, D. Lee et I. Essa, "Gaussian process regression flow for analysis of motion trajectories," dans *2011 International Conference on Computer Vision*. IEEE, 2011, p. 1164–1171.

[36] J. Joseph *et al.*, "A bayesian nonparametric approach to modeling motion patterns," *Autonomous Robots*, vol. 31, n°. 4, p. 383, 2011.

[37] G. S. Aoude *et al.*, "Threat assessment design for driver assistance system at intersections," dans *13th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2010, p. 1855–1862.

[38] D. Helbing et P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, n°. 5, p. 4282, 1995.

[39] Y. Yoo *et al.*, "Visual path prediction in complex scenes with crowded moving objects," dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, p. 2668–2677.

[40] A. Lawitzky *et al.*, "Interactive scene prediction for automotive applications," dans *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, p. 1028–1033.

[41] J. Firl et Q. Tran, "Probabilistic maneuver prediction in traffic scenarios." dans *ECMR*, 2011, p. 89–94.

[42] D. Meyer-Delius, C. Plagemann et W. Burgard, "Probabilistic situation recognition for vehicular traffic scenarios," dans *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, p. 459–464.

[43] T. Gindele, S. Brechtel et R. Dillmann, "Learning driver behavior models from traffic observations for decision making and planning," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, n°. 1, p. 69–79, 2015.

[44] A. Vemula, K. Muelling et J. Oh, "Modeling cooperative navigation in dense human crowds," dans *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, p. 1685–1692.

[45] A. Robicquet *et al.*, "Learning social etiquette : Human trajectory understanding in crowded scenes," dans *European conference on computer vision*. Springer, 2016, p. 549–565.

[46] P. Coscia *et al.*, "Long-term path prediction in urban scenarios using circular distributions," *Image and Vision Computing*, vol. 69, p. 81–91, 2018.

[47] A. Alahi *et al.*, "Social lstm : Human trajectory prediction in crowded spaces," dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, p. 961–971.

[48] F. Bartoli *et al.*, "Context-aware trajectory prediction," dans *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, p. 1941–1946.

[49] D. Varshneya et G. Srinivasaraghavan, "Human trajectory prediction using spatially aware deep attention models," *arXiv preprint arXiv :1705.09436*, 2017.

[50] N. Deo et M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, p. 1468–1476.

[51] A. Gupta *et al.*, "Social gan : Socially acceptable trajectories with generative adversarial networks," dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, p. 2255–2264.

[52] N. Radwan et W. Burgard, "Effective interaction-aware trajectory prediction using temporal convolutional neural networks."

[53] H. Manh et G. Alaghband, "Scene-lstm : A model for human trajectory prediction," *arXiv preprint arXiv :1808.04018*, 2018.

[54] T. Fernando *et al.*, “Soft+ hardwired attention : An lstm framework for human trajectory prediction and abnormal event detection,” *Neural networks*, vol. 108, p. 466–478, 2018.

[55] A. Vemula, K. Muelling et J. Oh, “Social attention : Modeling attention in human crowds,” dans *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, p. 1–7.

[56] A. Sadeghian *et al.*, “Car-net : Clairvoyant attentive recurrent network,” dans *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, p. 151–167.

[57] S. Huang *et al.*, “Deep learning driven visual path prediction from a single image,” *IEEE Transactions on Image Processing*, vol. 25, n°. 12, p. 5892–5904, 2016.

[58] U. Baumann *et al.*, “Predicting ego-vehicle paths from environmental observations with a deep neural network,” dans *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, p. 1–9.

[59] S. Hoermann, M. Bach et K. Dietmayer, “Dynamic occupancy grid prediction for urban autonomous driving : A deep learning approach with fully automatic labeling,” dans *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, p. 2056–2063.

[60] H.-S. Jeon, D.-S. Kum et W.-Y. Jeong, “Traffic scene prediction via deep learning : Introduction of multi-channel occupancy grid map as a scene representation,” dans *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, p. 1496–1501.

[61] A. Graves, “Generating sequences with recurrent neural networks,” *arXiv preprint arXiv :1308.0850*, 2013.

[62] D. P. Kingma *et al.*, “Semi-supervised learning with deep generative models,” dans *Advances in neural information processing systems*, 2014, p. 3581–3589.

[63] S. Becker *et al.*, “Red : A simple but effective baseline predictor for the trajnet benchmark,” dans *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, p. 0–0.

[64] S. Pellegrini *et al.*, “You’ll never walk alone : Modeling social behavior for multi-target tracking,” dans *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, p. 261–268.

[65] A. Lerner, Y. Chrysanthou et D. Lischinski, “Crowds by example,” dans *Computer graphics forum*, vol. 26, n°. 3. Wiley Online Library, 2007, p. 655–664.

[66] B. Zhou, X. Wang et X. Tang, “Understanding collective crowd behaviors : Learning a mixture model of dynamic pedestrian-agents,” dans *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, p. 2871–2878.

[67] B. Fisher, “Edinburgh informatics forum pedestrian database,” 2015.

[68] K. Gregor *et al.*, “Draw : A recurrent neural network for image generation,” *arXiv preprint arXiv :1502.04623*, 2015.

[69] M. Mirza et S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv :1411.1784*, 2014.

[70] J. Bergstra et Y. Bengio, “Random search for hyper-parameter optimization,” *Journal of Machine Learning Research*, vol. 13, n°. Feb, p. 281–305, 2012.

[71] L. Prechelt, “Early stopping-but when ?” dans *Neural Networks : Tricks of the trade*. Springer, 1998, p. 55–69.

[72] M. Arjovsky, S. Chintala et L. Bottou, “Wasserstein gan,” *arXiv preprint arXiv :1701.07875*, 2017.

ANNEXE A RÉCAPITULATIF DES PERFORMANCES DES APPROCHES MENTIONNÉES DANS LA REVUE DE LITTÉRATURE

Le Tableau A.1 présente le contexte d'évaluation des différents articles mentionnés dans la revue de littérature. Il précise :

- Les datasets utilisés.
- Les unités des résultats rapportés.
- Le protocole d'évaluation.

Tableau A.1 Datasets,unités et protocoles d'évaluation par article de la revue de littérature

	Dataset	Unité	Protocole d'évaluation
social-lstm [47]	eth/ucy	meters	1
context-aware [48]	ucy(zara)	meters	2
desire [11]	sdd	pixels(résolution 1/5)	1
carnet [56]	sdd	pixels	3
spatially-aware [49]	sdd(gates1,2,3)	meters	-1
red [63]	trajnet benchmark	meters	3
soft-hard [54]	NY Grand Central	meters	2
soft-hard	Edinburgh Informatic	meters	2
social-gan [51]	eth/ucy	meters	1
scene-lstm [53]	eth/ucy	meters	4
cnn [5]	eth/ucy	meters	1
sophie [6]	eth/ucy	meters	1
sophie	sdd(piétons)	pixels	3
iatcnn [52]	eth/ucy	meters	1

La légende de la colonne protocole d'évaluation est la suivante :

— Cas -1 : Le protocole n'est pas mentionné.

— Cas 1 : Validation croisée avec omission de scène. On entraîne sur un sous-ensemble des scènes de l'ensemble de donnée et on teste sur la ou les scènes restantes. On recommence pour chaque permutation.

— Cas 2 : On conserve 80% des données de chaque scène pour l'entraînement et 20% des données de chaque scène pour le test.

— Cas 3 : On entraîne sur un sous-ensemble des scènes de l'ensemble de donnée et on teste sur la ou les scènes restantes.

— Cas 4 Comme 1. Le modèle est ré-entrainé sur 50% des données de la scène d'évaluation et testé sur les 50 % restant.

Tableau A.2 Performances reportées par approche et par article (première partie)

	social-forces	lstm	s2s	social-lstm	context-axare	social-lstm	desire-it0-best	desire-it4-10%	car-net	spatially-aware
social-lstm [47]	0,39/0,60	0,44/0,98	-	0,27/0,61	-	-	-	-	-	-
context-aware [48]	-	1,40/	-	1,34/	1,22/	-	-	-	-	-
desire [11]	-	-	9,54	-	-	10,23	5,33	-	-	-
carnet [56]	36,48/58,14	-	-	31,19/56,97	-	35,73/63,35	-	25,72/51,80	-	-
spatially-aware [49]	-	-	-	0,15/0,33	-	-	-	-	0,11/0,25	0,38/1,30
red [63]	0,41/1,39	1,100/3,114	0,390/1,331	0,675/2,098	-	-	-	-	-	-
soft-hard [54]	3,364/5,80	1,99/4,52	-	-	-	-	-	-	-	-
soft-hard	3,124/3,909	1,524/2,510	-	-	-	-	-	-	-	-
social-gan [51]	-	0,79/1,59	-	0,70/1,52	-	-	-	-	-	-
scene-lstm [53]	-	0,23/0,21	-	0,25/0,22	-	-	-	-	-	-
cnn [5]	-	0,70/1,52	-	0,72/1,54	-	-	-	-	-	-
sophie [6]	-	0,70/1,52	-	0,72/1,54	-	-	-	-	-	-
sophie	36,48/58,14	-	-	31,19/56,97	-	-	19,25/34,05	25,72/51,8	-	-
iatcnn [52]	0,39/0,60	0,29/0,93	-	0,27/0,61	-	-	-	-	-	-

Tableau A.3 Performances reportées par approche et par article (deuxième partie)

	tcn	red	soft+harwired attention	sgan-20v20	scene-lstm	cnn	sophie	iatcnn
social-lstm	-	-	-	-	-	-	-	-
context-aware	-	-	-	-	-	-	-	-
desire	-	-	-	-	-	-	-	-
carnet	-	-	-	-	-	-	-	-
spatially-aware	0,364/1,229	-	-	-	-	-	-	-
red	-	1,096/3,011	-	-	-	-	-	-
soft-hard	-	0,986/1,311	-	-	-	-	-	-
soft-hard	-	-	-	-	-	-	-	-
social-gan	-	-	0,58/1,18	-	-	-	-	-
scene-lstm	-	-	-	0,07/0,07	-	-	-	-
cnn	-	-	0,58/1,18	-	0,59/1,22	0,54/1,15	-	-
sophie	-	-	0,58/1,18	-	-	0,54/1,15	-	-
sophie	-	-	27,246/41,440	-	-	16,27/29,38	-	-
iatcnn	-	-	0,58/1,18	-	-	0,54/1,15	0,19/0,27	-

Dans le tableau, on retrouve en colonne le nom du modèle proposé dans l'étude en question. En ligne on présente les performances de tous les modèles mentionnés ou utilisés comme référence par l'étude concernée. Pour chaque étude, l'ensemble de données utilisé est mentionné dans le tableau A.1. On peut remarquer dans ce tableau que la hiérarchie en termes de performances entre différents modèles n'est pas constante d'une étude à l'autre. De plus les performances d'un même modèle ne sont pas constantes d'une étude à l'autre. Cette variabilité est due entre autre aux différents protocoles de validation observés ainsi qu'au soin apporté à l'entraînement des modèles. On remarque aussi que les performances de tous les modèles sont globalement comprises dans un intervalle de quelques dizaines de centimètres ou pixels. A une telle échelle il est difficile de conclure si un modèle est vraiment meilleur qu'un autre ou si il a été simplement mieux entraîné.

ANNEXE B DÉFINITION DE LA MÉTRIQUE COMPLÉMENTAIRE POUR ÉVALUER LA PRISE EN COMPTE DES INTERACTIONS SPATIALES PAR LES MODÈLES

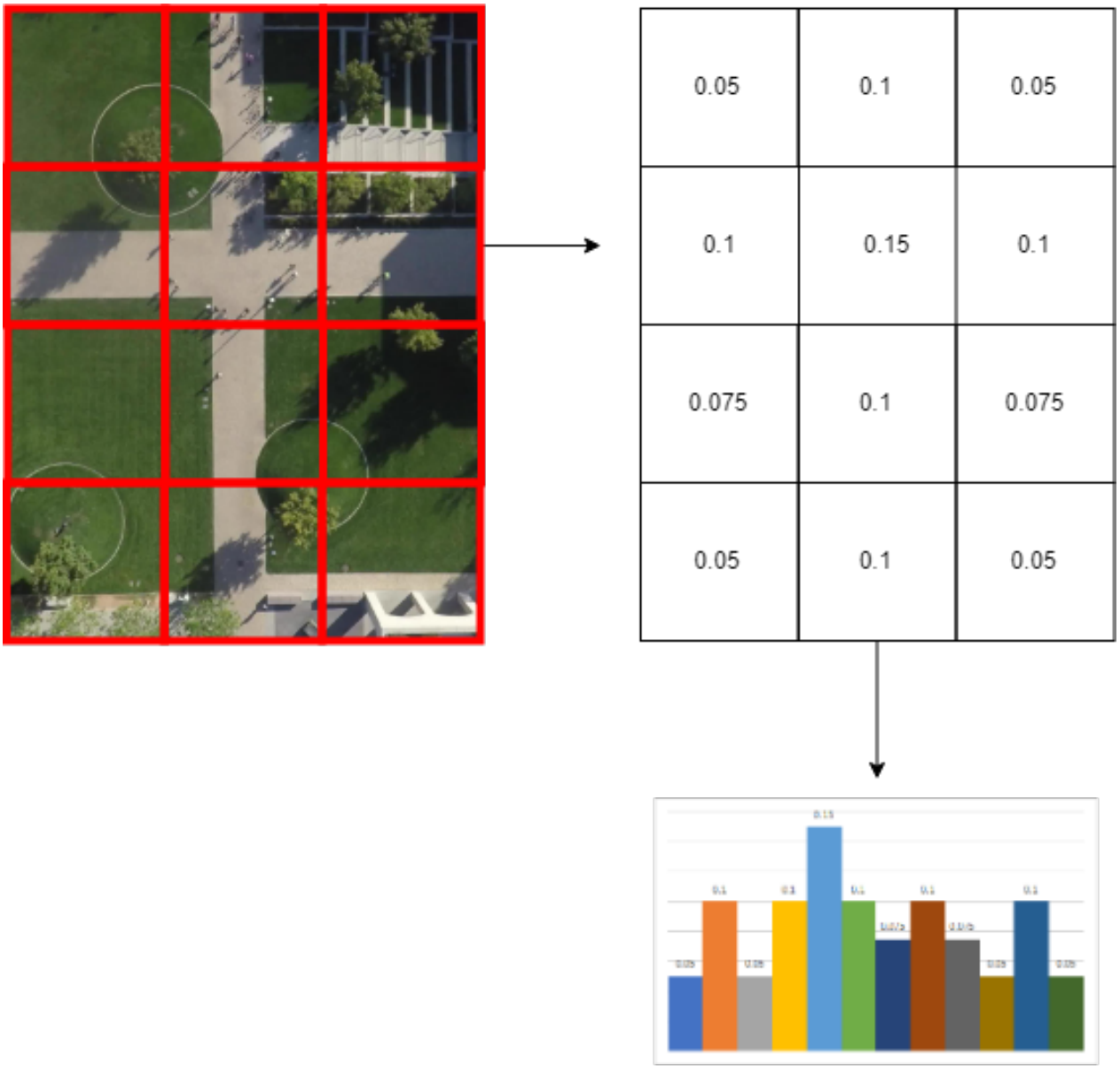


Figure B.1 Calcul de l'histogramme pour la métrique spatiale complémentaire

On divise la vue aérienne de chaque scène (dans sa représentation en mètres) en une grille de cellules carrées toutes de même dimensions. Pour un modèle, on effectue l'ensemble

des prédictions de l'ensemble de test. On obtient ainsi l'ensemble des positions prédites par ce modèle. On compte ensuite pour chaque cellule de la grille le nombre de ces points qui la traverse. On divise ces résultats par le nombre total de points prédits pour obtenir la proportion par cellule. On aplatit cette grille afin d'obtenir un histogramme. L'histogramme obtenu permet de caractériser la distribution spatiale des positions des agents apprise par ce modèle. On effectue de même cette opération pour les positions réelles observées dans l'ensemble de test. L'histogramme obtenu permet de caractériser la manière dont les agents se déplacent réellement dans la scène. Pour évaluer la capacité d'un modèle à apprendre correctement la manière réelle dont les agents arpentent la scène, on calcule la distance L1 entre les deux histogrammes (la distance de L1 entre deux histogrammes ayant chacun n barres est exprimée dans l'équation B.1). On répète le processus pour toutes les scènes de l'ensemble de test. Le résultat final de la métrique correspond à la moyenne des distances sur toutes les scènes. À partir de cette distance, on pourra comparer les différents modèles. On peut faire varier la taille du côté des cellules pour étudier différents niveaux de granularité. Le processus d'obtention de l'histogramme est représenté sur la figure B.1.

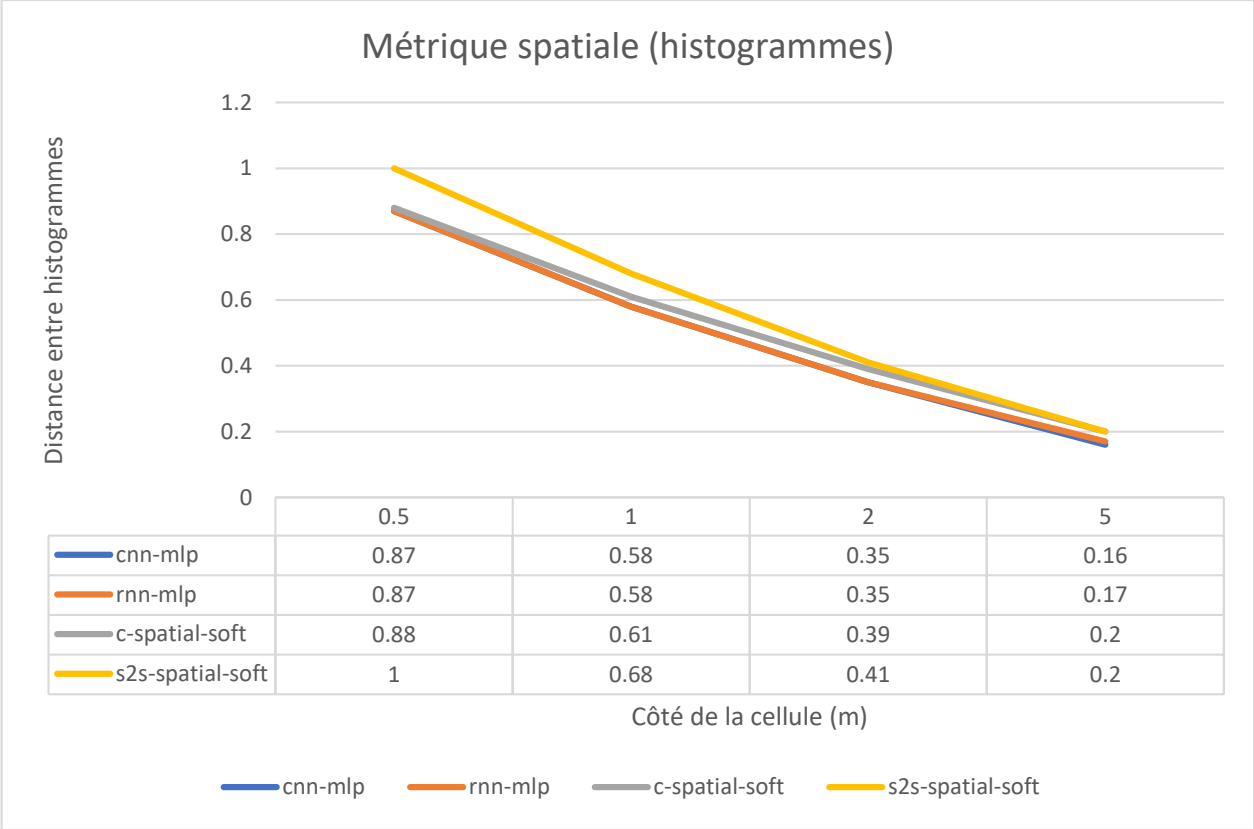
$$L_1(p, q) = \sum_{i=1}^{n} |p_i - q_i| \tag{B.1}$$

ANNEXE C RÉSULTATS COMPLÉMENTAIRES POUR LA MÉTRIQUE COMPLÉMENTAIRE POUR ÉVALUER LA PRISE EN COMPTE DES INTERACTIONS SPATIALES PAR LES MODÈLES

On calcule la métrique spatiale complémentaire pour quatre modèles :

- *cnn-mlp* : le modèle de base naïf utilisant un simple *CNN*.
- *rnn-mlp* : le modèle de base naïf utilisant un *LSTM*.
- *s2s-spatial* : le modèle de base attentif utilisant un *LSTM* et un module de *soft-attention* pour les interactions spatiales.
- *c-spatial-soft* : le modèle de base attentif utilisant un *CNN* et un module de *soft-attention* pour les interactions spatiales.

On calcule la métrique pour différentes tailles de côté des cellules : 0.5 m, 1 m, 2 m et 5 m. Les résultats sont représentés sur la figure C.1.



	0.5	1	2	5
cnn-mlp	0.87	0.58	0.35	0.16
rnn-mlp	0.87	0.58	0.35	0.17
c-spatial-soft	0.88	0.61	0.39	0.2
s2s-spatial-soft	1	0.68	0.41	0.2

Figure C.1 Résultats pour la métrique complémentaire pour les interactions spatiales

Sur la figure C.1, les deux courbes inférieures correspondent aux deux modèles ne prenant pas en compte les interactions spatiales. Les courbes des deux modèles avec attention spatiale ne traversent pas celles des modèles naïfs. Les modèles naïfs sont donc légèrement meilleurs selon cette métrique que les modèles avec attention spatiale. Cette métrique ne permet donc pas de mettre en évidence un impact de l'attention spatiale sur la prise en compte des interactions spatiales par les modèles. Ce résultat corrobore celui de la métrique principale pour les interactions spatiales utilisée dans le mémoire.